TESIS – TE142599

SISTEM PERINGATAN KEDATANGAN KERETA API DI PERLINTASAN MENGGUNAKAN SENSOR ACCELEROMETER DAN NEURAL NETWORK

HERI ARDIANSYAH
NRP. 07111450040009

DOSEN PEMBIMBING
Dr. Muhammad Rivai, ST., MT.
Ir. Luhur Prihadi Eka Nurabdi, MT.

PROGRAM MAGISTER
BIDANG KEAHLIAN ELEKTRONIKA
DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI ELEKTRO
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2018

TESIS – TE142599

SISTEM PERINGATAN KEDATANGAN KERETA API DI PERLINTASAN MENGGUNAKAN SENSOR ACCELEROMETER DAN NEURAL NETWORK

HERI ARDIANSYAH
NRP. 07111450040009

DOSEN PEMBIMBING
Dr. Muhammad Rivai, ST., MT.
Ir. Luhur Prihadi Eka Nurabdi, MT.

PROGRAM MAGISTER
BIDANG KEAHLIAN ELEKTRONIKA
DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI ELEKTRO
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2018

LEMBAR PENGESAHAN

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Teknik (M.T)
di
Institut Teknologi Sepuluh Nopember

oleh:

Heri Ardiansyah
NRP. 07111450040009

Tanggal Ujian : 19 Desember 2017
Periode Wisuda : Maret 2018

Disetujui oleh:

1. Dr. Muhammad Rivai, ST., MT. (Pembimbing I)
   NIP: 19690426 199403 1 003
2. Ir. Luhur Prihadi Eka Nurabdi, MT. (Pembimbing II)
   NIP: 19690412 199403 1 010
3. Ronny Mardiyanto, ST., MT., Ph.D. (Penguji)
   NIP: 19810118 200312 1 003
4. Astria Nur Irfansyah, ST., M.Eng. (Penguji)
   NIP: 19810325 201012 1 002
5. Muhammad Attamimi, B.Eng, M. Eng, PhD. (Penguji)
   NPP: 1985 2017 11039

Dekan Fakultas Teknologi Elektro,

Dr. Tri Arief Sardjono, S.T., M.T.
NIP. 19700212 199512 1 001

Halaman ini sengaja dikosongkan

PERNYATAAN KEASLIAN TESIS

Dengan ini saya menyatakan bahwa isi keseluruhan Tesis saya dengan judul "**SISTEM PERINGATAN KEDATANGAN KERETA API DI PERLINTASAN MENGGUNAKAN SENSOR ACCELEROMETER DAN NEURAL NETWORK**" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, Desember 2017

Heri Ardiansyah
NRP. 07111450040009

Halaman ini sengaja dikosongkan

SISTEM PERINGATAN KEDATANGAN KERETA API DI PERLINTASAN MENGGUNAKAN SENSOR ACCELEROMETER DAN NEURAL NETWORK

Nama Mahasiswa : Heri Ardiansyah
NRP : 07111450040009
Pembimbing : 1. Dr. Muhammad Rivai, ST., MT.
2. Ir. Luhur Prihadi Eka Nurabdi, MT.

ABSTRAK

Banyaknya perlintasan kereta api tanpa palang pintu otomatis dan petugas penjaga dapat meningkatkan resiko kecelakaan. Kondisi ini tidak dapat dielakkan karena adanya perluasan lingkungan di sekitar wilayah jalur kereta api, seperti komplek perumahan, pertokoan dan perkembangan aktifitas perekonomian. Oleh karena itu, sistem peringatan kedatangan kereta api di perlintasan menjadi kebutuhan solusi yang mendesak. Setiap kereta api yang sedang berjalan akan mengakibatkan getaran pada lintasan rel yang dilewatinya. Pada penelitian ini digunakan sebuah sensor accelerometer untuk membaca getaran pada rel agar dapat mendeteksi kedatangan kereta api. Pada sistem ini menggunakan sensor accelerometer MPU6050 dan 16-bit Analog to Digital Converter sehingga dapat menghasilkan rentang pengukuran sebesar 16384 LSB/g (-32768 ~ 32768). Kekuatan getaran yang terukur menjadi data untuk memperkirakan posisi kedatangan kereta api. Sinyal getaran dalam domain waktu tersebut dikonversi menjadi spektrum frekuensi menggunakan Fast Fourier Transform. Algoritma Neural Network digunakan untuk mengenali pola frekuensi getaran rel yang disebabkan oleh kereta api yang dapat dibedakan dari sepeda motor, mobil, dan truk yang melewati perlintasan. Mikrokontroler Arduino Uno digunakan untuk membaca data sensor dan mengirimkannya ke komputer. Hasil pengujian menunjukkan bahwa getaran kereta api dapat dideteksi pada jarak terjauh hingga 74 meter dan dapat mengenali pola getaran kereta api terhadap yang lainnya dengan tingkat keberhasilan sebesar 100%. Dengan dirancangnya sistem peringatan dini kedatangan kereta api diharapkan keselamatan pengguna jalan dapat lebih ditingkatkan terutama pada daerah perlintasan kereta api.

Kata Kunci : accelerometer, getaran rel, *neural network*, perlintasan kereta, pola frekuensi.

Halaman ini sengaja dikosongkan

TRAIN ARRIVAL WARNING SYSTEM AT RAILROAD CROSSING USING ACCELEROMETER SENSOR AND NEURAL NETWORK

By : Heri Ardiansyah
Student Identity Number : 07111450040009
Supervisors : 1. Dr. Muhammad Rivai, ST., MT.
2. Ir. Luhur Prihadi Eka Nurabdi, MT.

ABSTRACT

The number of railroad crossing without warning system and official guard can increase the risk of accidents. The increasing number cannot be controlled because of the expansion of neighbour near the railtracks, such citizen houses, markets, and other economy activities which reveal new road crossing the railtrack. Thus, an early warning to alert of a train arrival become an urgent solution indeed. A running train will generate a vibration to the railtracks. In this experiment, an accelerometer sensor is used to read the railtrack vibration to detect the train arrival. The design of the system is using an MPU6050 with 16-bit Analog to Digital Converter to create a measurement range with 16384 LSB/g sensitivity (-32768 ~ 32768). The vibration amplitudes will be use to predict the position of arriving train. The time domain vibration first converted to the frequency domain using a Fast Fourier Transformation. Next, a Neural Network algorithm will be utilized to identify the frequency spectrum pattern of the arriving train from motorcycles, cars, or any trucks passing the railroad. An Arduino Uno microcontroller will be equipped to transfer the sensor data into the computer. The experiment results that the train vibration can be detected in the fartest distance upto 74 meters and the frequency patterns can be recognized from other with accuracy 100%. By designing a train arrival warning system there will achieve the safe of people and the train around the railroad crossing.

Keywords : accelerometer, frequency pattern, neural network, railroad crossing, rail vibration

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Alhamdulillah, puji dan syukur penulis panjatkan kepada Allah SWT, karena atas ridlo-Nya lah tesis ini dapat diselesaikan. Tesis berjudul "**Sistem Peringatan Kedatangan Kereta Api di Perlintasan menggunakan Sensor Accelerometer dan *Neural Network***" ini disusun untuk memenuhi sebagai persyaratan memperoleh gelar Magister Teknik (MT) pada Jurusan Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember.

Penulis menyadari bahwa dalam penyusunan tesis ini tidak terlepas dari bantuan berbagai pihak. Oleh karena itu, dengan ketulusan dan kerendahan hati penulis menyampaikan terima kasih kepada:

1. Bapak Dr. Muhammad Rivai, ST., MT. dan Ir. Luhur Prihadi Eka Nurabdi, MT. selaku Dosen Pembimbing yang telah banyak memberikan saran, bantuan, serta sabar dalam membimbing penulis.
2. Bapak Ronny Mardiyanto, ST., MT., Ph.D., Bapak Astria Nur Irfansyah, ST., M.Eng., Ph.D., dan Bapak Muhammad Attamimi, B.Eng, M. Eng, PhD. selaku Dosen Penguji Ujian Sidang Tesis atas saran dan masukannya.
3. Bapak Dr. Eng. Ardyono Priyadi, ST., M.Eng. selaku ketua Jurusan Teknik Elektro.
4. (Almh) Ibunda Suci Lasmiati dan Ayahanda M. Maksum tercinta, atas segala dukungan dan doanya hingga terselesaikannya tesis ini.
5. Pihak lain yang tidak dapat penulis sebutkan satu per satu.

Penulis menyadari bahwa tesis ini masih belum sempurna, oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun. Penulis berharap semoga tesis ini dapat bermanfaat bagi pengembangan ilmu pengetahuan dan teknologi serta bagi masyarakat.

Surabaya, 3 Desember 2017
Heri Ardiansyah

Halaman ini sengaja dikosongkan

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB 1
PENDAHULUAN

1.1. **Latar Belakang**

Peningkatan jumlah perlintasan jalan kereta api tidak dapat dielakkan karena adanya perkembangan lingkungan di sekitar wilayah jalur kereta api, seperti munculnya komplek perumahan, komplek pertokoan dan perkembangan aktifitas perekonomian warga setempat. Kondisi ini menjadi krusial apabila dikaitkan dengan pembangunan jalur ganda (double track). Menghadapi tantangan ke depan adalah terkait peningkatan keselamatan di pintu perlintasan.

PT KAI menyebutkan bahwa pada tahun 2016 tercatat angka kecelakaan di pintu perlintasan di wilayah Divre I Sumatra Utara tercatat sebanyak 52 kasus (PR KAI, 2017). Tingginya biaya pengadaan palang pintu dan keterbatasan sumber daya manusia sebagai penjaga baik oleh pihak PT. KAI atau pemerintah setempat juga menjadi kendala keselamatan di perlintasan liar. Oleh karena itu dibutuhkan suatu cara lain untuk mengatasi limitasi keadaan tersebut.

Suatu kereta api yang sedang berjalan menjadi pembangkit gelombang getaran pada rel yang dilewatinya. Sepanjang jalur rel jika diamati bunyinya maka pada jarak beberapa ratus meter pun akan terdengar suara getaran yang menandakan rel telah bergetar meskipun dengan tingkat kekuatan getaran yang rendah. Getaran yang terjadi pada rel akan dibaca oleh sensor accelerometer untuk memperkirakan sampai sejauh mana jarak yang dapat dibaca sensor ini. Jarak deteksi ini akan berguna untuk dijadikan sinyal peringatan kedatangan bahwa kereta api sudah mendekat.

Penggunaan sensor accelerometer telah banyak dilakukan pada beberapa penelitian. Pada Tabel 1.1 terdapat pengelompokan sensor accelerometer yang banyak digunakan pada penelitian terhadap getaran rel sebagai sinyal peringatan akan kedatangan kereta api, dengan perkiraan teoritisnya yang mampu mendeteksi dengan jangkauan sejauh 800 meter dari pusat getaran.

Tabel 1.1 Penggunaan Sensor Sebagai Detektor Getaran Rel (Santos dkk., 2013)

Sensor Technology/ Method	Detects Passing	Stationary Detection	Measures Distance	Provides Direction	Measures Length	Measures Speed	Intrusive Installation	Requires LOS	Approach Detection	Early Detection	Max Theoretical Distance
Treadle Mechanism	X			X[a]			X		X		
Inductive Sensor (Rail Module)	X			X		X	X		X		
Infrared (IR)	X	X[b]		X[c]	X[d]				X		
Reflectometry (TDR)	X	X	X			X			X	X	<3mi
Accelerometer	X		X[e]						X	X	<0.5mi
Magnetometer (AMR)	X	X		X	X	X			X		
Doppler Radar	X			X		X		X	X	X[f]	<2mi
Pulse-Doppler	X	X	X	X	X	X		X	X	X[f]	<2mi
FMCW Radar	X	X	X	X	X	X		X	X	X[f]	<1mi
Acoustic (Horn Only)	X			X					X	X	<0.5mi

Pada penelitian-penelitian sebelumnya, sistem deteksi getaran dengan memanfaatkan sensor MEMS accelerometer 3-axis ADXL345 diaplikasikan untuk mendeteksi getaran yang disebabkan oleh kereta yang melewati posisi sensor (Jiang dkk., 2014) (Wang dkk.,2006). Sedangkan sistem yang menggunakan crystal piezoelectric PCB 352B telah digunakan untuk mendeteksi keberadaan kereta sejauh 800 meter dari posisi sensor (Angrisani dkk., 2010).

Dengan menggunakan sensor MEMS accelerometer, diharapkan dapat dirancang suatu sistem peringatan dini yang dapat mendeteksi posisi kereta dari jarak tertentu yang cukup memadai.

1.2. Perumusan Masalah

Beberapa permasalahan analisa getaran rel kereta di perlintasan adalah sebagai berikut:

1. Bagaimana sensitifitas sensor accelerometer MEMS mendeteksi getaran rel hingga jarak tertentu.
2. Bagaimana mengubah bentuk sinyal getaran dengan transformasi fourier.
3. Bagaimana mengidentifikasi sinyal getaran rel yang disebabkan oleh kereta api menggunakan neural network.

1.3. Tujuan Dan Manfaat

Tujuan dari penelitian ini adalah untuk mengetahui keberhasilan deteksi sensor MEMS accelerometer untuk mengindentifikasi getaran rel jauh sebelum kereta api datang. Secara terperinci tujuan penelitian ini adalah:

1. Mengetahui sensitifitas sensor accelerometer terhadap getaran rel kereta hingga jarak tertentu.
2. Mengetahui metode pengubahan sinyal getaran menggunakan transformasi fourier.
3. Mengetahui cara identifikasi sinyal getaran menggunakan neural network.

Dari hasil penelitian ini akan diperoleh manfaat antara lain untuk memperoleh data mengenai karakteristik getaran rel yang disebabkan oleh kereta yang mendekat. Dan dari identifikasi karakteristik yang diketahui dapat dirancang suatu sistem peringatan dini kedatangan kereta api.

Halaman ini sengaja dikosongkan

BAB 2
KAJIAN PUSTAKA DAN DASAR TEORI

2.1 Sistem Peringatan menggunakan Accelerometer

Penelitian menggunakan sensor accelerometer untuk mendeteksi getaran rel kereta api telah dilakukan menggunakan accelerometer *piezo crystal*. Tipe accelerometer ini membutuhkan konverter analog ke digital agar proses komputasi getaran dapat dilakukan. Spesifikiasi pengukurannya ditentukan oleh spesifkasi komponen accelerometer sebesar 25K sampel per detik. Dan hasil perhitungan frekuensi spektrum dominan berada pada daerah frekuensi 1KHz ~ 1.3KHz dan 9.5 ~ 9.7KHz. Eksperimen tersebut dilakukan pada suatu jalur kereta dengan kondisi terdapat sekumpulan pekerja yang melakukan pekerjaan perawatan terhadap jalur kereta itu (Angrisani dkk., 2010). Dengan kondisi tersebut, jika sensor atau suatu sistem peringatan dipasangkan pada suatu perlintasan, maka akan didapatkan manfaat yang lebih besar untuk banyak kalangan/masyarakat.

2.2 Sistem Peringatan pada Perlintasan Kereta Api Indonesia

Perlintasan sebidang antara jalur kereta dengan jalan raya merupakan jenis perlintasan dimana rel kereta berada pada satu bidang lintasan dengan jalan raya yang dijaga oleh seorang petugas jaga lintasan (PJL). Pada perlintasan resmi yang dibangun oleh PT. KAI, seperti terlihat pada Gambar 2.1a, terdapat perangkat pengamanan yang harus dipasangkan sebagai peringatan dini kedatangan kereta api adalah:

1. Rambu-rambu/marka berupa silang dan huruf KA.
2. Palang pintu penutupan otomatis atau manual.
3. Peringatan dini untuk petugas jaga lintasan.
4. Peringatan dini untuk pengguna jalan raya.

Untuk memberikan sinyal peringatan dini akan kedatangan kereta api pada perlintasan, PT KAI menggunakan sistem deteksi secara manual dengan cara mengelola persinyalan kereta api yakni melalui perangkat *Westrace* seperti dalam Gambar 2.2.

(a) (b)

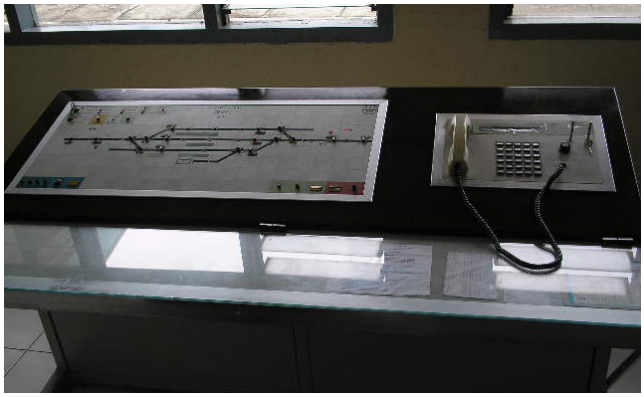
Gambar 2.1 Perlintasan Kereta Api (wikimapia.org, 2013)
(a) Perlintasan Sebidang (b) Gardu Perlintasan

Dengan perangkat ini, petugas PJL pada gardu perlintasan dalam Gambar 2.1b akan menerima sinyal komunikasi dari stasiun terdekat menandakan adanya kereta api yang akan lewat.

Sistem persinyalan *Westrace* dibagi menjadi beberapa bagian, yaitu:

1. LCP mempunyai fungsi sebagai pengendali seluruh kegiatan yang ada di emplasemen stasiun ditampilkan secara visual di panel berupa indikator dan tombol-tombol perintah.
2. Interlocking berfungsi sebagai otak pengolah data segala perintah dari LCP dan deteksi dari peralatan luar melalui relay-relay interface.
3. Catu daya berfungsi sebagai power supply bagi seluruh peralatan internal/eksternal, terdiri dari jaringan PLN, genset dan batere.
4. Peralatan komunikasi (Local communication console) berguna untuk komunikasi dengan stasiun sebelah, pusat pengendali ataupun dengan gardu pintu perlintasan. Di dalam perangkat komunikasi juga terdapat hubungan blok yang memungkinkan hanya ada satu rangkaian kereta dalam satu section blok.
5. Peralatan luar/ekternal terdiri dari wesel (pemindah jalur kereta api), sinyal lampu (pemberi tanda apakah kereta api boleh berjalan atau tidak), *track circuit* dan *axle counter* (pendeteksi keberadaan kereta api).

Prinsip kerja peralatan sistem persinyalan *Westrace* adalah dengan cara operator melakukan instruksi melalui penekan tombol, kemudian instruksi tersebut diolah oleh modul DIP (*Digital Input Parallel*) diumpankan ke suatu modul *scanner* SCN41.



Gambar 2.2 Perangkat Kendali *Westrace* pada Stasiun KA (Santoso, 2014)

2.3 Sarana & Prasarana Kereta Api

Kereta api adalah bentuk transportasi rel yang terdiri dari serangkaian kendaraan yang ditarik sepanjang jalur kereta api untuk mengangkut kargo atau penumpang. Gaya gerak rangkaian kereta api disediakan oleh lokomotif yang terpisah atau motor individu dalam beberapa unit. Ada berbagai jenis kereta api yang dirancang untuk tujuan tertentu. Kereta api bisa terdiri dari kombinasi satu atau lebih dari lokomotif dan gerbong kereta terpasang, atau beberapa unit yang digerakkan sendiri (atau kadang-kadang pelatih bertenaga tunggal atau diartikulasikan, disebut sebuah kereta mobil).

2.3.1 Kereta Penumpang

Kereta penumpang digunakan untuk mengantarkan penumpang pergi dari suatu tempat ke tempat lainnya. Kereta penumpang biasanya dapat dibagi menjadi dua operasi: kereta api antar kota dan kereta api lokal seperti dalam Gambar 2.3a. Untuk kereta api antar kota memerlukan kecepatan yang lebih tinggi hingga maksimal 120 km/jam, rute lama, dan frekuensi yang lebih rendah yang biasanya terjadwal. Sedangkan kereta api lokal melibatkan kecepatan yang relatif rendah, rute pendek, dan frekuensi yang lebih tinggi terutama pada saat jam sibuk.

(a) (b)

Gambar 2.3 Kereta Penumpang
(a) Kereta Penumpang Jarak Jauh (b) Kereta Komuter

Kereta antar kota adalah kereta api jarak jauh yang beroperasi dengan beberapa stasiun antar kota. Kereta biasanya memiliki fasilitas seperti gerbong makan. Beberapa kereta api jarak jauh diberi nama khusus. Kereta regional adalah kereta api jarak menengah yang menghubungkan kota dengan daerah terpencil, daerah sekitarnya, atau memberikan layanan regional, sehingga sering berhenti dan memiliki kecepatan yang relatif rendah.

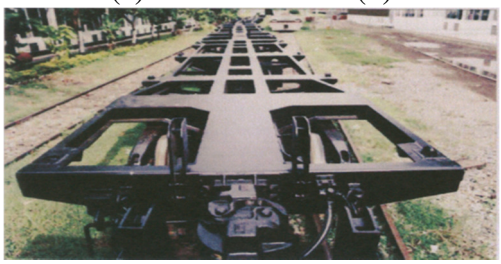
Kereta komuter, seperti dalam Gambar 2.3b, melayani pinggiran perkotaan menyediakan layanan komuter sehari-hari. Dalam kota besar biasanya tersedia juga jaringan kereta bandara dengan akses cepat dari pusat kota ke bandara. Kereta ini memiliki spesifikasi kecepatan rata-rata 40 km/jam atau kecepatan maksimum 70 km/jam.

2.3.2 Kereta Barang

Kereta api barang atau Kereta Api Bagasi adalah kereta api yang digunakan untuk mengangkut barang (kargo), pupuk, hasil tambang (pasir, batu, batubara ataupun mineral), ataupun kereta api trailer gerbong datar yang digunakan untuk mengangkut peti kemas yang terlihat dalam rangkaian kereta Gambar 2.4a. Selain itu digunakan gerbong khusus untuk mengangkut ternak ataupun tangki untuk mengangkut minyak atau komoditas cair lainnya seperti dalam gambar 2.4b. Berat kosong (maksimum) satu gerbong datar yang terlihat dalam Gambar 2.4c memiliki bobot sebesar 14.500 kg, sedangkan berat kosong kereta tangki sebesar 20.000 kg.

(a)

(b)



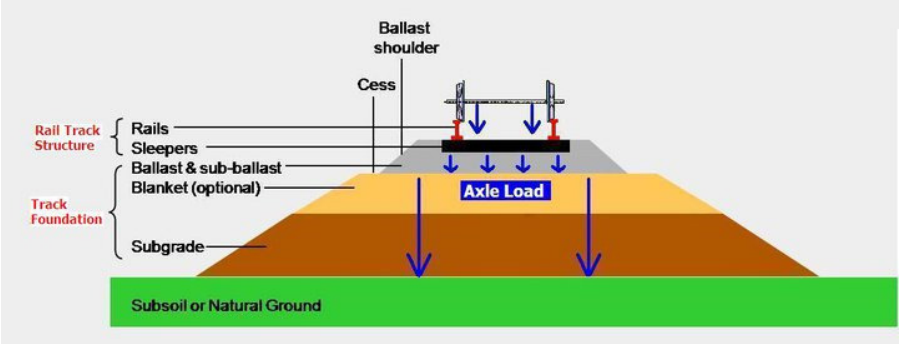
(c)

Gambar 2.4 Kereta Barang
(a) Kereta Peti Kemas (b) Kereta Tangki (c) Gerbong Datar

2.3.3 Konstruksi Rel Kereta Api

Kereta api berjalan dengan roda besi, sehingga membutuhkan jalan khusus agar dapat berjalan dengan baik yang juga terbuat dari besi/baja. Jalan baja ini memiliki karakteristik dan syarat-syarat khusus yang berbeda dengan jalan aspal, sehingga konstruksinya lebih rumit dan melibatkan banyak komponen. Jalan rel kereta api harus dibangun dengan kokoh, karena setiap rangkaian kereta yang lewat memiliki beban yang berat yang akan dilalui berulang kali oleh beberapa rangkaian kereta. Oleh karena itu, konstruksi rel kereta dibuat sebaik mungkin agar mampu menahan beban berat atau disebut Beban Gandar (*Axle Load*) dari rangkaian kereta yang berjalan di atasnya, sehingga jalan baja ini dapat bertahan dalam waktu yang lama dan memungkinkan rangkaian kereta dapat berjalan dengan cepat, aman, dan nyaman.

Lapisan Ballast merupakan suatu lapisan berupa batu-batu berukuran kecil yang ditaburkan di bawah trek rel, tepatnya di bawah, samping, dan sekitar bantalan rel (*sleepers*). Bahkan terkadang dijumpai bantalan rel yang “tenggelam” tertutup lapisan *ballast*, sehingga hanya terlihat batang relnya saja.



Gambar 2.5 Konstruksi Rel Kereta Api

Merujuk pada bagan di atas, pada dasarnya konstruksi jalan rel KA terdiri atas 2 bagian. Bagian bawah adalah *Track Foundation* atau Lapisan Landasan/Pondasi, dan bagian atas adalah *Rail Track Structure* atau Struktur Trek Rel. Prinsipnya, jalan rel kereta api harus dapat mentransfer tekanan yang diterimanya dengan baik yang berupa beban berat (*axle load*) dari rangkaian kereta yang melintas. Dalam arti, jalan rel keretaharus tetap kokoh ketika dilewati rangkaian kereta api, sehingga rangkaian kereta dapat melintas dengan cepat, aman, dan nyaman. Roda-roda kereta api yang melintas akan memberikan tekanan berupa beban berat ke permukaan trek rel. Oleh batang rel (*rails*), tekanan tersebut diteruskan ke bantalan (*sleepers*) yang ada dibawahnya. Lalu, dari bantalan akan diteruskan ke lapisan ballast dan *sub-ballast* di sekitarnya. Oleh lapisan *ballast*, tekanan dari bantalan ini akan disebar ke seluruh permukaan tanah disekitarnya, untuk mencegah amblesnya trek rel.

Fungsi lapisan *ballast* adalah:

1. untuk meredam getaran trek rel saat rangkaian kereta melintas,
2. menyebarkan *axle load* dari trek rel ke lapisan landasan di bawahnya, sehingga trek rel tidak ambles,
3. menjaga trek rel agar tetap berada di tempatnya,
4. sebagai lapisan yang mudah direlokasi untuk menyesuaikan dan meratakan ketinggian trek rel (*Levelling*),
5. memperlancar proses drainase air hujan,
6. mencegah tumbuhnya rumput yang dapat mengganggu drainase air hujan.

2.4 Prinsip Getaran dan Gelombang

Getaran adalah gerak bolak-balik benda secara teratur melalui titik keseimbangan. Salah satu ciri getaran adalah adanya amplitudo (simpangan terbesar suatu getaran). Sedangkan gelombang adalah getaran yang merambat pada suatu medium. Gelombang terjadi karena terdapat sumber getaran. Pada perambatanya getaran merambatkan energi gelombang, sedangakan medium perantaranya tidak ikut merambat. Getaran dan gelombang melibatkan satuan-satuan periode, frekuensi, dan amplitudo getaran

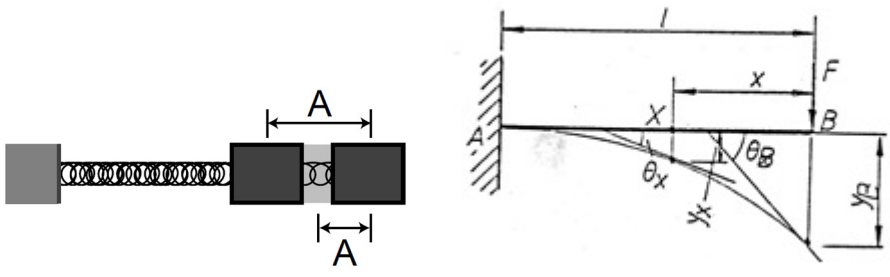
Periode atau waktu yang dibutuhkan benda untuk melakukan satu kali getaran dinyatakan dalam satuan detik, melalui persamaan:

Periode getaran (T) = waktu getar / Jumlah getaran (n)

Sedangkan ferkuensi yang merupakan jumlah getran dalam satu sekon dengan satuan Hertz (Hz) dapat dinyatakan sebagai:

Frekuensi (f) = Jumlah getaran / Waktu getaran (t)

Amplitudo adalah besarnya simpangan suatu getaran terjadi dalam satuan jarak (meter). Dalam hal ini simpangan yang terukur adalah dalam satuan mekanis. Satuan amplitudo juga memiliki 2 macam dari posisi simpangannya. Yang pertama adalah amplitudo yangdiukur dari posisi tengah ke posisi ektstrim kiri atau kanan. Sedangkan yang kedua adalah amplitudo puncak-ke-puncak yang diukur dari satu posisi ekstrim ke posisi ekstrim yang berlawanan.



(a) (b)

Gambar 2.6 Prinsip Getaran (a) Amplitudo Getaran (b) Balok Kantilever

Balok kantilever adalah sebuah balok yang memiliki perletakan (*support*) hanya pada salah satu ujungnya. Kantilever panjang l, menerima beban titik F di ujung, seperti dalam Gambar 2.6b, dikenai momen inersia I pada penampangnya dan modulus elastisitas E. Untuk menganalisa perpindahan (*displacement*) balok kantilever ini dilihat potongan titik X, jaraknya x dari ujung (titik B), 0<x<1.

Momen di titik X

$$M_x = -Fx \tag{2.1}$$

$$M_b = EI\frac{d^2y}{dx^2} \tag{2.2}$$

Untuk menentukan sudut lentur. Karena,

$$\frac{dy}{dx} = \frac{-Fx^2}{2EI} + C_1 \tag{2.3}$$

dan pada x=0 atau di titik B, $\frac{dy}{dx}$ tidak sama dengan 0,

sedangkan pada x=1 atau di titik A, $\frac{dy}{dx} = 0$, menghasilkan

$$\frac{dy}{dx} = \theta_x = \frac{-Fx^2}{2EI} + \frac{Fl^2}{2EI}$$

Sehingga, sudut lentur maksimum yang terjadi pada x=0, di B, adalah

$$\theta_B = \frac{-F0^2}{2EI} + \frac{Fl^2}{2EI} = \frac{Fl^2}{2EI} \tag{2.4}$$

Lenturan di titik x, persamaan 9.13 diintegralkan, hasilnya seperti berikut.

$$y_x = \frac{-Fx^3}{6EI} + \frac{Fl^2x}{2EI} + C_2 \tag{2.5}$$

Untuk menentukan konstanta C2, pada x=0 atau di titik B, lenturannya tidak sama dengan nol dan pada x=1 atau di titik A, $Y_A = 0$, sehingga

$$0 = \frac{-Fl^3}{6EI} + \frac{Fl^2l}{2EI} + C_2$$

atau,

$$C_2 = \frac{-Fl^3}{3EI} \tag{2.6}$$

Jadi:

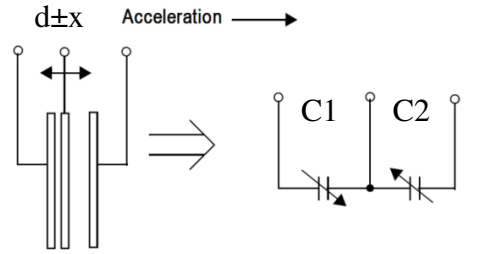
$$y_x = \frac{-Fl^3}{6EI} + \frac{Fl^2x}{2EI} + \frac{-Fl^3}{3EI} \tag{2.7}$$

Lenturan maksimum yang terjadi pada x=0, di B adalah,

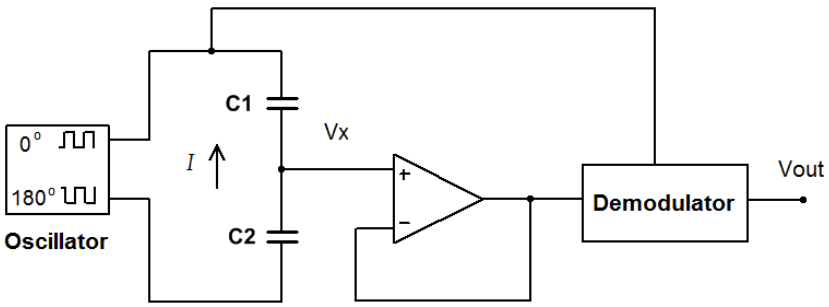
$$y_x = 0 + 0 - \frac{Fl^3}{3EI} = \frac{-Fl^3}{3EI} \quad \text{mm} \tag{2.8}$$

2.5 Prinsip Kerja MEMS Accelerometer

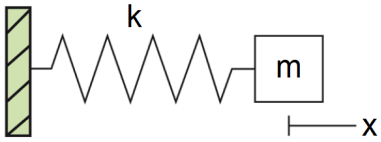
Sensor accelerometer MEMS (*micro electro mechanical system*) melakukan deteksi akselerasi menggunakan prinsip kapasitansi sehingga mengeluarkan tegangan berdasarkan perbedaan jarak antar plat kapasitornya. Seperti terlihat pada Gambar 2.7a, pada salah satu atau kedua plat ini ditambahkan arus listrik Dimensi kapasitor pembentuknya menjadi sangat prinsip untuk perhitungan nilai kapasitansi pada rangkaian ekivalennya.. Perubahan gap antar plat mengubah nilai kapasitansi yang terukur sebagai tegangan keluaran.



(a)



(b)



(c)

Gambar 2.7 Prinsip Kerja Accelerometer
(a) Kapasitor (b) Rangkaian Ekivalen (c) Sistem Pegas

Suatu nilai kapasitansi C dapat dinyatakan dalam Persamaan 2.9 sebagai berikut:

$$C = \varepsilon_a \frac{A}{d} \tag{2.9}$$

Dimana:

C = kapasitansi (F)

ε_a= konstanta dielektrik bahan (F/m)

A = luas permukaan (m^2)

d = jarak dielektrik (m)

Pada Gambar 2.4a, masing-masing nilai kapasitansi adalah sebesar:

$$C_1 = C_2 = \varepsilon_r \frac{A}{d} = \varepsilon_a \frac{1}{d} \tag{2.10}$$

Jika x adalah perubahan jarak dielektrik (d ± x), maka:

$$\left.\begin{aligned} C_1 &= \varepsilon_a \frac{1}{d+x} = C_0 - \Delta C \\ C_2 &= \varepsilon_a \frac{1}{d-x} = C_0 - \Delta C \end{aligned}\right\} C_1 + \quad C_2 = 2C_0 \tag{2.11}$$

$$C_2 - C_1 = (C_0 + \Delta C) - (C_0 - \Delta C) = \varepsilon_a \frac{1}{d-x} - \varepsilon_a \frac{1}{d+x}$$

$$2\Delta C = \varepsilon_a \frac{2x}{d^2 - x^2} \tag{2.12}$$

$$\rightarrow \Delta C x^2 + \varepsilon_a x - \Delta C d^2 = 0$$

Jika $x \rightarrow 0$, maka

$$x = \frac{d^2}{\varepsilon_a} \tag{2.13}$$

$$x = d\frac{\Delta C}{C_0} \tag{2.14}$$

$$\frac{\Delta C}{C_0} = \frac{x}{d} \tag{2.15}$$

Pada Gambar 2.4b, karena tegangan supply MEMS berupa tegangan bolak balik sehingga ada beda fasa 180°, maka Vx dapat diturunkan menjadi:

$$I = \frac{V_0 - V_x}{X_{c2}} = \frac{V_x - (-V_0)}{X_{c1}} \tag{2.16}$$

$$(V_0 - V_x)C_2 = (V_x + V_0)C_1 \tag{2.17}$$

$$V_x = \left(\frac{C_2 - C_1}{C_2 + C_1}\right)V_0$$

$$V_x = \frac{2\Delta C}{2C_0}V_0 \tag{2.18}$$

$$V_x = \frac{x}{d}V_o \tag{2.19}$$

$$x = \frac{d}{V_0}V_x \tag{2.20}$$

Dengan menggunakan prinsip pegas seperti pada Gambar 2.4c, ketika pegas bergeser sebesar x, maka kapasitor juga bergeser sebesar x, sehingga:

$$ma = kx \tag{2.21}$$

Didapatkan percepatan:

$$a = \frac{k}{m}\frac{d}{V_0}V_x \tag{2.22}$$

2.6 Transformasi Fourier FFT

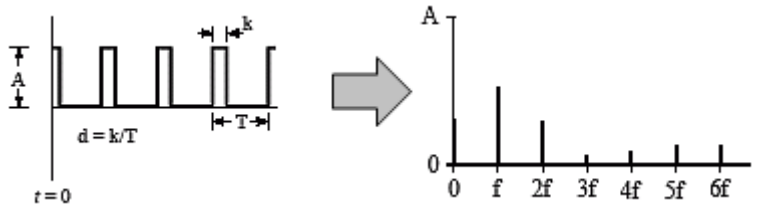
Analisa Fourier merupakan salah satu teknik dalam matematika yang berdasar pada dekomposisi sinyal ke dalam sinusoidal. Dari transformasi ini didapatkan data dalam domain frekuensi.

$$X(f) = \int_{-\infty}^{\infty} x(t).\, e^{-j\omega t}\, dt \tag{2.23}$$

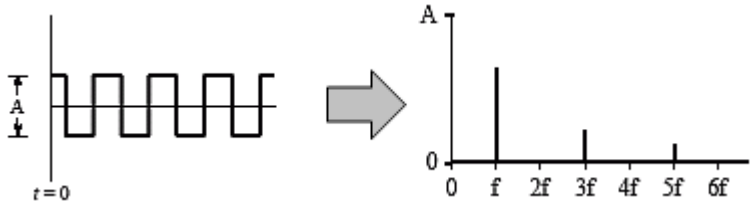
Transformasi Fourier sangat sulit untuk diaplikasikan pada pengukuran sebenarnya, sehingga digunakan pendekatan yang mampu merepresentasikan transformasi Fourier secara digital yaitu *Discrete Fourier Transform* (DFT), dinyatakan:

$$X(k) = \sum_{n=0}^{N-1} x[n].e^{-\frac{j2\pi nk}{N}}, 0 \leq k \leq N-1 \qquad (2.24)$$

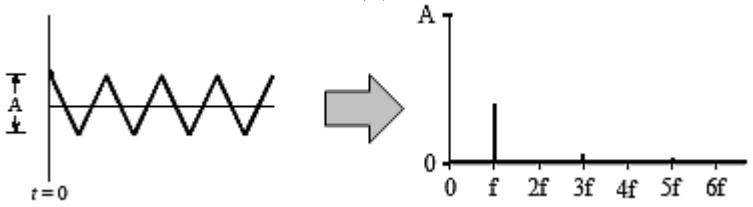
Fast Fourier Transform (FFT) adalah algoritma yang digunakan untuk menghitung DFT dengan proses perhitungan yang lebih efisien.. Dalam Gambar 2.8 ditunjukkan beberapa bentuk transformasi sinyal domain waktu ke dalam domain frekuensi.



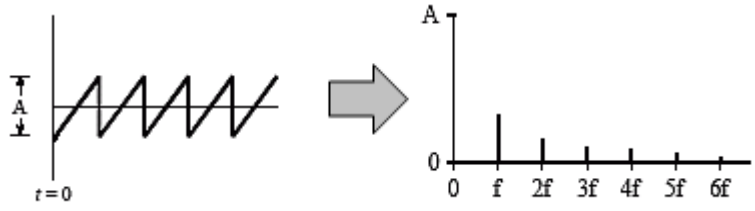
(a)



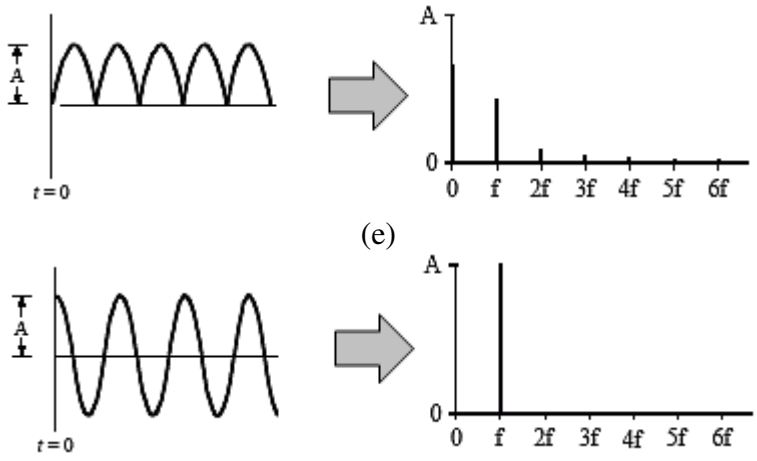
(b)



(c)



(d)



Gambar 2.9 FFT Sinyal-sinyal Domain Waktu ke Frekuensi
(a) Sinyal Pulsa (b) Sinyal Logika (c) Sinyal Segitiga
(d) Sinyal Gergaji (e) Sinyal Disearahkan (f) Sinyal Sinusoida

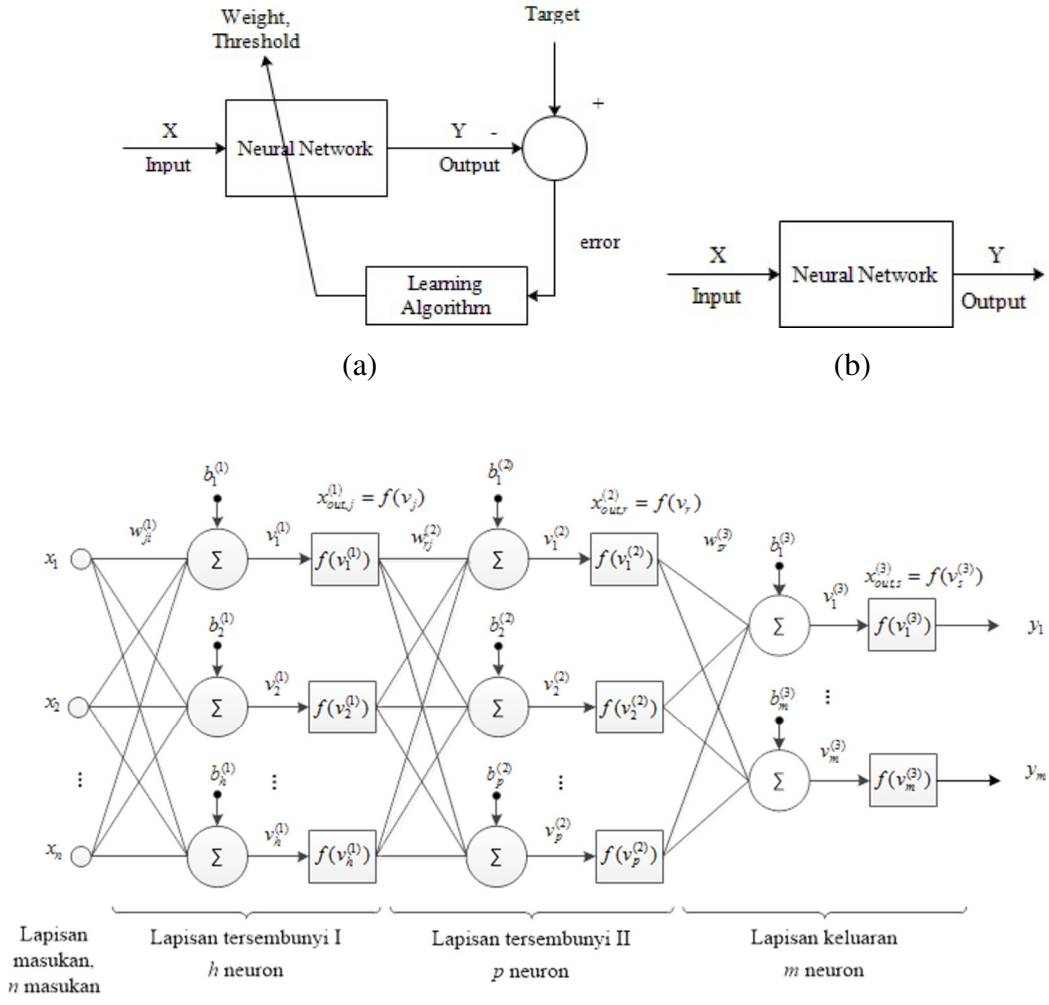
2.7 Artificial Neural Network

ANN terdiri dari elemen-elemen sederhana yang meniru sistem saraf biologis manusia. Salah satu algoritma dari ANN yang digunakan sebagai identifikasi pola frekuensi adalah alogoritma pelatihan *backpropagation*. Algoritma ini akan memproses input X untuk menghasilkan output Y sesuai nilai yang diinginkan. Pemrosesan kedua matriks X dan Y (persamaan 2.11-12) merupakan tahap feed forward input X hingga mencapai output Y (Gambar 2.12a).

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \dots x_{1,j} \\ x_{2,1} & x_{2,2} & \dots x_{2,j} \\ \vdots & \vdots & \ddots \ \vdots \\ x_{i,1} & x_{i,2} & \dots x_{i,j} \end{bmatrix} \tag{2.25}$$

$$Y = \begin{bmatrix} y_{1,1} & y_{1,2} & \dots y_{1,j} \\ y_{2,1} & y_{2,2} & \dots y_{2,j} \\ \vdots & \vdots & \ddots \ \vdots \\ y_{i,1} & y_{i,2} & \dots y_{i,j} \end{bmatrix} \tag{2.26}$$

Ketika output dari ANN tidak sama dengan output yang diinginkan, maka *error* dari keduanya akan dihitung dan digunakan untuk memperbaiki nilai *weight* dan *threshold* sebelumnya. Perbaikan dilakukan dari layer output ke layer sebelumnya hingga layer input. Perbaikan pada weight dan threshold disebut tahap *backward.* Setelah perbaikan mencapai layer input, maka proses *feed forward* akan berjalan kembali. Proses tersebut akan berulang sampai error yang diinginkan tercapai.



(a) (b)

(c)

Gambar 2.10 Skema Neural Network
(a) Tahap backward (b) Tahap feed forwardk
(c) Multilayer Perceptron

2.7.1 Algoritma Feedforward

Pada Gambar 2.10b ditunjukan tahap *feed forward*. Adapun tahapan algoritmanya sebagai berikut

1. Tentukan banyaknya masukan x (n masukan), lapisan tersembunyi pertama (h neuron), lapisan tersembunyi kedua (p neuron), dan lapisan keluaran y.
2. Tentukan target *error* (E_{target}) yang diinginkan, target (d), *slope* fungsi aktifasi (α) dan laju pelatihan μ_0 .

3. Inisialisasi bobot ($w_{ji}^{(1)}$, $w_{rj}^{(2)}$ dan $w_{sr}^{(3)}$) dan bias ($b_j^{(1)}$, $b_r^{(2)}$ dan $b_s^{(3)}$) secara acak, hal ini hanya dilakukan untuk iterasi pertama (k=1), untuk iterasi selanjutnya menggunakan hasil perbaikan *weight* dan bias dari algoritma *backpropagation*.
4. Hitung $v_j^{(1)}$ dari $j = 1,2,...,h$ dengan Persamaan 2.27,

$$v_j^{(1)} = b_j^1 + \sum_{i=1}^{n} x_i w_{ji}^{(1)} \tag{2.27}$$

5. Hitung $x_{out,j}^{(1)}$ dengan rumus $x_{out,j}^{(1)} = f(v_j^{(1)})$, dimana $f(v_j^{(1)})$ merupakan fungsi aktivasi, pada kasus klasifikasi menggunakan fungsi aktifasi *binary sigmoid*, seperti pada Persamaan 2.28,

$$f(v_j) = \frac{1}{1+e^{-2\alpha v_j}} \tag{2.28}$$

6. Hitung $v_r^{(2)}$ dari $r = 1,2,..., p$, kemudian hitung $x_{out,r}^{(2)}$

$$\begin{aligned} v_r^{(2)} &= b_r^{(2)} + \sum_{j=1}^{h} x_{out,j}^{(1)} w_{rj}^{(2)} \\ x_{out,r}^{(2)} &= f(v_r^{(2)}) \\ &= \frac{1}{1+e^{-2\alpha v_r^{(2)}}} \end{aligned} \tag{2.29}$$

7. Hitung $v_s^{(3)}$ dari $s = 1,2,...,m$, kemudian hitung $x_{out,s}^{(3)}$,

$$\begin{aligned} v_s^{(3)} &= b_s^{(3)} + \sum_{r=1}^{p} x_{out,r}^{(2)} w_{sr}^{(3)} \\ x_{out,s}^{(3)} &= f(v_s^{(3)}) \\ &= \frac{1}{1+e^{-2\alpha v_s^{(3)}}} \end{aligned} \tag{2.30}$$

8. $x_{out,s}^{(3)}$ pada Persamaan 2.30 merupakan keluaran dari proses *feedforward*, *y*.

2.7.2 Algoritma Backpropagation

Pada Gambar 2.10b ditunjukan tahap *backpropagation*. Adapun algoritmanya adalah sebagai berikut:

1. Lakukan tahap *feedforward.*
2. Hitung E dengan Persamaan 2.17.

$$E = \frac{1}{2}\sum_{s=1}^{m}(d_s - y_s)^2 \tag{2.31}$$

3. Jika nilai $E > E_{target}$ (belum konvergen), lakukan perbaikan terhadap nilai bobot dan bias tiap lapisan kemudian lakukan tahap *feedforward* mulai dari langkah 4. Jika konvergen maka proses *backpropagation* berhenti.
4. Jika menggunakan algoritma LMS (*Least Mean Square*), hitung μ dengan Persamaan 2.14, dengan $\mu_0 > 0$ dan $\tau >> 1$. Jika tidak menggunakan algoritma LMS maka nilai $\mu = \mu_0$,

$$\mu = \frac{\mu_0}{1+\frac{k}{\tau}} \tag{2.32}$$

5. Hitung perbaikan nilai bobot dan bias pada lapisan keluaran,

$$w_{sr}^{(3)}(k+1) = w_{sr}^{(3)}(k) + \mu\delta_s^{(3)}x_{xout,r}^{(2)} \tag{2.33}$$

$$b_s^{(3)}(k+1) = b_s^{(3)}(k) + \mu\delta_s^{(3)} \tag{2.34}$$

dimana nilai $\delta_s^{(3)}$ dihitung menggunakan Persamaan 2.35,

$$\delta_s^{(3)} = (d_s - x_{out,s}^{(3)})g(v_s^{(3)}) \tag{2.35}$$

dengan $g(v_s^{(3)})$ merupakan $\frac{df(v_j)}{dv_j}$, sehingga $g(v_s^{(3)})$ dapat dihitung menggunakan Persamaan 2.36.

$$g(v_s^{(3)}) = \alpha f(v_s^{(3)})[1 - f(v_s^{(3)})] \tag{2.36}$$

6. Hitung perbaikan nilai bobot dan bias pada lapisan tersembunyi II,

$$w_{rj}^{(2)}(k+1) = w_{rj}^{(2)}(k) + \mu\delta_r^{(2)} x_{xout,j}^{(1)} \tag{2.37}$$

$$b_r^{(2)}(k+1) = b_r^{(2)}(k) + \mu\delta_r^{(2)}$$

dimana nilai $\delta_r^{(2)}$ adalah pada Persamaan 2.21.

$$\delta_r^{(2)} = \left(\sum_{s=1}^{m} \delta_s^{(3)} w_{sr}^{(3)} \right) \alpha f\left(v_r^{(2)}\right)\left[1 - f\left(v_r^{(2)}\right)\right] \tag{2.38}$$

7. Hitung perbaikan nilai bobot dan bias pada lapisan tersembunyi I,

$$w_{ji}^{(1)}(k+1) = w_{ji}^{(1)}(k) + \mu\delta_j^{(1)} x_i$$

$$b_j^{(1)}(k+1) = b_j^{(1)}(k) + \mu\delta_j^{(1)}$$

dimana nilai $\delta_j^{(1)}$ dihitung menggunakan Persamaan 2.17.

$$\delta_j^{(1)} = \left(\sum_{r=1}^{p} \delta_r^{(2)} w_{rj}^{(2)} \right) \alpha f\left(v_j^{(1)}\right)\left[1 - f\left(v_j^{(1)}\right)\right] \tag{2.39}$$

Halaman ini sengaja dikosongkan

BAB 3
METODOLOGI PENELITIAN

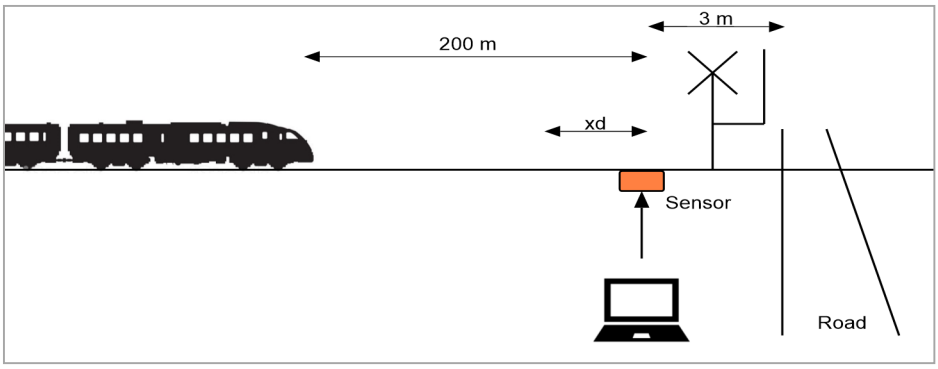
3.1 Studi Lapangan

Lokasi penelitian bertempat di perlintasan sebidang di Kecamatan Cerme, Kabupaten Gresik, Jawa Timur, atau pada posisi peta -7.2231093 Lintang Selatan, 112.5690513 Bujur Timur.

Pada Gambar 3.1 diatas, untuk kondisi lalu lintas normal, waktu menutup palang pintu perlintasan dari kondisi terbuka ke kondisi tertutup sempurna seperti Gambar 3.1b dibutuhkan waktu sekitar 12 detik. Pada detik ke-12 ini, ketika kereta api lewat jalan raya sudah terbilang aman. Apabila kecepatan kereta api tersebut sebesar 60 km/jam atau 16.7 m/detik, maka sebelumnya, posisi kereta ketika palang perlintasan masih terbuka adalah pada jarak 200 meter. Untuk lalu lintas agak ramai, waktu yang dibutuhkan menutup palang mencapai 16 detik. sehingga jarak kereta dari perlintasan menjadi sejauh 267 meter. Akan tetapi pada lalu lintas sepi, waktu yang dibutuhkan menutup palang hanya 8 detik. Atau posisi aman perlintasan dari kereta pada jarak lebih dekat yakni pada 133 meter dari perlintasan.

(a) (b)

Gambar 3.1 Kondisi Palang Pintu Perlintasan (a) membuka (b) menutup



Gambar 3.2 Diagram Pengambilan Data Getaran

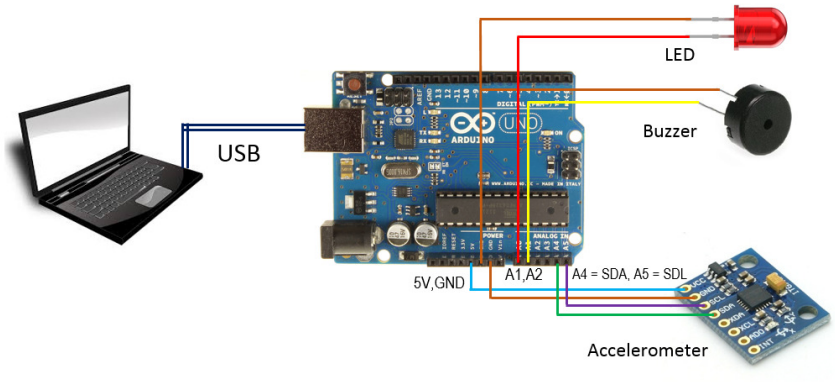
Dari ketiga kondisi pemantauan ini, rata-rata dibutuhkan waktu 12 detik untuk mempersiapkan lalu lintas di perlintasan atau pada posisi kereta 200 meter dari perlintasan. Sehingga dalam diagram perancangan Gambar 3.2 digunakan jarak pengambilan data dimulai pada 200 meter dari kedatangan kereta api.

3.2 Perancangan hardware dan software

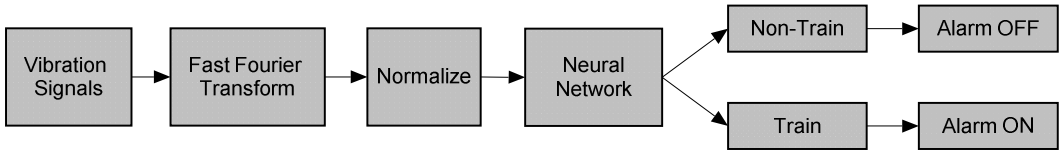
Chip MPU 6050 tersemat pada modul pcb (printed circuit board) mendapat supply dari board mikrokontroller Arduino sebesar 5V, meskipun pada spesifikasinya dapat diberikan supply 3.3V. Kecepatan transfer data dari mikrokontroler Arduino ke komputer dilakukan setup sebesar 115200 bps melalui USB Serialcom untuk mengakomodasi 1000 data per detik yang diterima dari sensor accelerometer.

Dalam Gambar 3.3, koneksi data accelerometer ke mikrokontroler Arduino menggunakan komunikasi I2C pada pin SCL-SDA yakni pin A4 dan A5. Sedangkan pin A1 untuk menyalakan LED dan pin A2 untuk membunyikan buzzer.

Dalam Gambar 3.4, perancangan software atau perangkat lunak terdiri atas 3 blok pengolahan data getaran. Yang pertama adalah blok FFT, kemudian blok Normalisasi, dan blok Neural Network yang memliki fungsi utama menyeleksi jenis getaran yang direkam.



Gambar 3.3 Koneksi Wiring Perancangan Hardware



Gambar 3.4 Diagram Perancangan Software

3.3 Pengambilan data sinyal getaran

Untuk mengambil sinyal getaran di lokasi penelitian, sensor ditempatkan pada posisi sesuai pada perancangan dalam Gambar 3.2. Modul sensor diikatkan di atas lempengan braket besi yang dikaitkan pada rel (Gambar 3.2a). Modul sensor ini dipasangkan pada posisi axis-x searah dengan arah grafitasi bumi seperti yang terlihat dalam Gambar 3.3b.

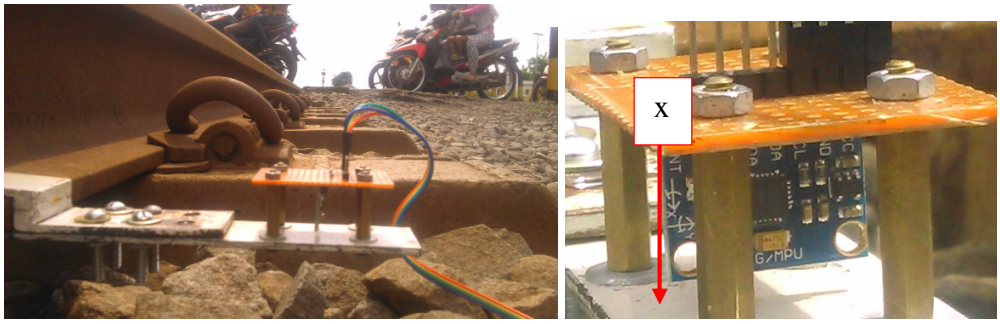
Masing-masing kondisi gelombang domain waktu dapat diilustrasikan dalam Gambar 3.5 dengan ciri-ciri sebagai berikut:

A. Gelombang No-Signal

   Pengambilan gelombang no-signal adalah pada kondisi ketika di lokasi tidak terdapat kendaraan yang melintasi jalan raya ataupun tidak terdapat kereta api yang sedang berjalan.

B. Gelombang Sepeda Motor

   Gelombang sepeda motor diambil ketika 1 buah sepeda motor melintasi jalan raya dan mengenai rel.



(a) (b)

Gambar 3.5 Pemasangan Sensor Accelerometer di Perlintasan
(a) Tampak samping (b) posisi axis-x

C. Gelombang Mobil

Gelombang mobil diambil ketika 1 buah mobil melintasi jalan raya dan mengenai rel.

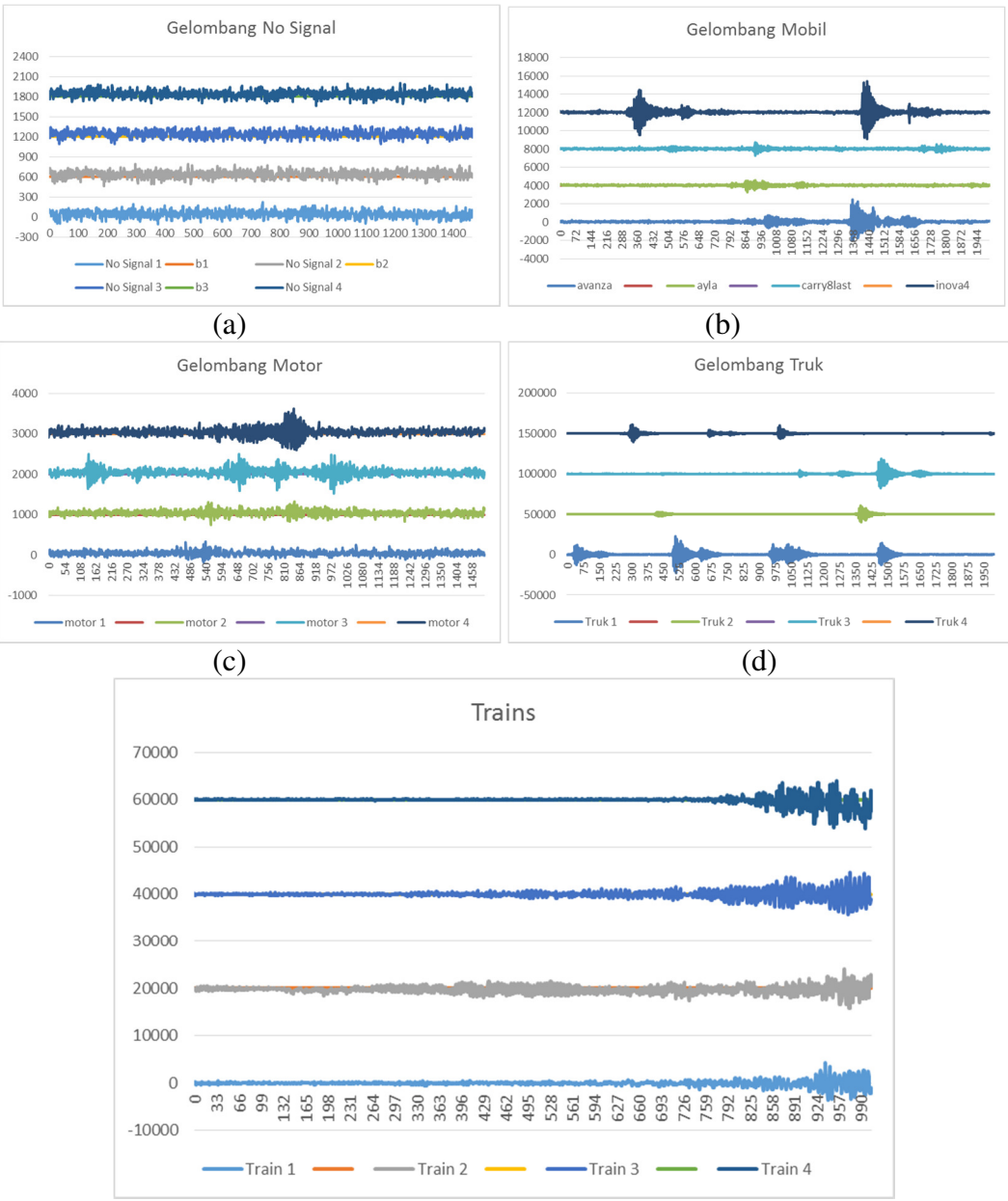
D. Gelombang Truk

Gelombang truk diambil ketika 1 buah truk melintasi jalan raya dan mengenai rel.

E. Gelombang Kereta Api

Gelombang kereta api diambil ketika kereta api mulai mendekati titik pengukuran dan dihentikan seketika melewati sensor.

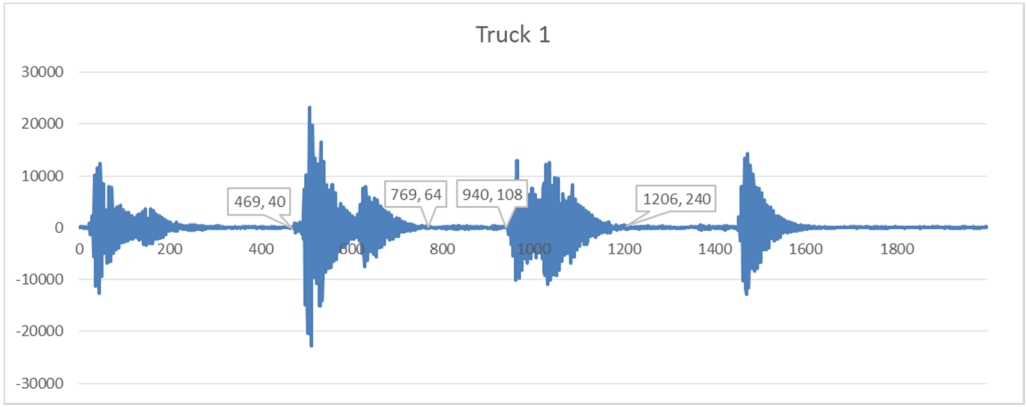
Amplitudo minimum dan maksimum, seperti terlihat pada Gambar 3.6, pada gelombang no-signal berkisar antara -200 ~ +200 LSB, pada gelombang motor antara +500 ~ -500 LSB, pada gelombang mobil antara +3000 ~ -3000, pada gelombang truk antara +10000 hingga -10000 LSB. Sedangkan gelombang kereta api memiliki simpangan terjauh sensor pada +32768 ~ -32768 LSB.



Gambar 3.6 Gelombang Domain Waktu.
(a) No Signal (b) Sepeda Motor (c) Mobil (d) Truk (e) Kereta

3.4 Penentuan Jarak Deteksi Kereta Api

Perbandingan durasi getaran dan tinggi amplitudo antara non-kereta dan kereta akan dijadikan penentuan jarak deteksi getaran kereta.



Gambar 3.7 Panjang Gelombang Truk

Tabel 3.1 Panjang Gelombang Truk

	Titik awal (ms)	Titik akhir (ms)	Panjang getaran (ms)
Titik 1	469	769	300
Titik 2	940	1206	266
Rata-rata			283

Salah satu kendaraan yang menghasilkan getaran diantaranya adalah truk yang memiliki amplitudo paling tinggi. Karena penentuan getaran kereta untuk deteksi jarak dengan cara membandingkan dengan pola gelombang selain kereta, maka dipilih pola gelombang truk, dalam Gambar 3.7, yang memiliki grafik gelombang paling jelas daripada gelombang motor ataupun gelombang mobil karena dari gelombang truk tersebut dapat dilihat titik awal dan titik akhir gelombang.

Dalam tabel 3.1, pola gelombang truk memiliki panjang getaran sekitar 300 ms, sedangkan panjang gelombang kereta api dipastikan lebih lebar dari itu. Sehingga panjang gelombang yang dapat digunakan untuk penentuan jarak deteksi kereta api yakni 2 kali dari panjang gelombang truk atau sebesar 600 ms, yang dalam penelitian ini diambil selama 1000 ms.

Amplitudo sinyal getaran no-signal seperti dalam Gambar 3.6a yang berkisar antara -200 ~ +200 LSB menandakan bahwa tidak ada getaran pada rel kereta. Ketika amplitudo melebihi *range* nilai tersebut, maka dapat diartikan bahwa ada obyek yang menggetarkan rel kereta.

3.5 Proses Transformasi FFT

Sinyal getaran yang dihasilkan oleh sensor accelerometer akan diubah dari fungsi waktunya menjadi fungsi frekuensi untuk mengetahui pola frekuensi masing-masing obyek baik kereta maupun non kereta.

Dilakukan tranformasi Fast Fourier menggunakan *script* dalam Microsoft Excel (Lungu, 2011). Untuk memastikan kalibrasi fungsi FFT ini, diberikan sinyal sinusoida sederhana pada dua frekuensi dari suatu data yakni f1 = 200 Hz dan f2 = 350 Hz, seperti persamaan 3.1 berikut:

$$y = 4\sin 2\pi 200t + 17\cos 2\pi 350t \tag{3.1}$$

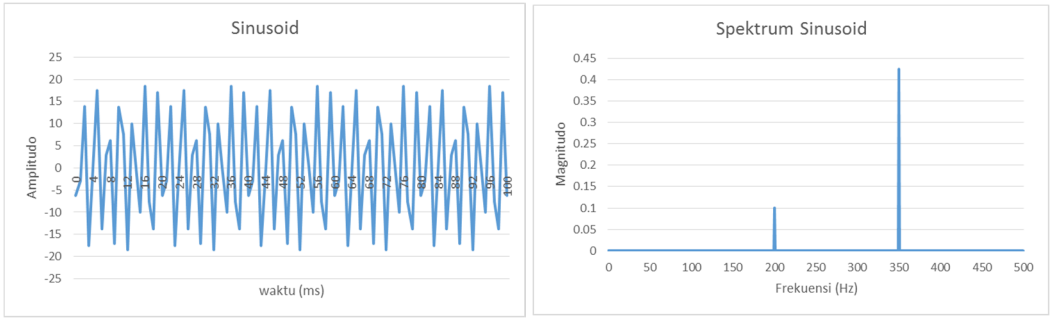
Sehingga spektrum frekuensi hasil kalibrasi transformasi FFT dapat diilustrasikan dalam Gambar 3.8b.

Modul accelerometer sebagai perangkat utama memiliki sample data maksimum 1000 data per 1 detik (Invensense, 2013), sehingga frekuensi sampling didapatkan sebesar 1000Hz. Data getaran kemudian diubah menjadi spektrum frekuensinya menggunakan FFT. Input FFT sebanyak 1000 data sehingga didapatkan spektrum frekuensi sebesar 500Hz.

$$\text{Frekuensi sampling (Fs)} = \frac{N}{T} = \frac{1000}{1} = 1000 \quad \text{Hz.} \tag{3.2}$$

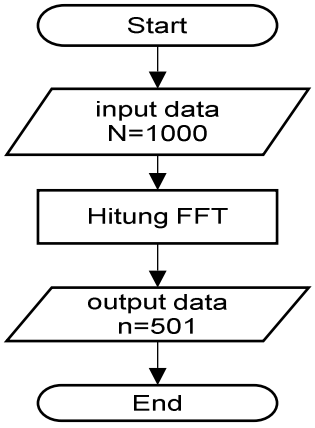
$$\text{Jadi frekuensi maksimum} \quad (F_{max}) = \frac{Fs}{2} = 500 \quad \text{Hz} \tag{3.3}$$

$$\text{Resolusi frekuensi (dF)} = \text{Fs/N} = 1000/1000 = 1 \text{ Hz} \tag{3.4}$$

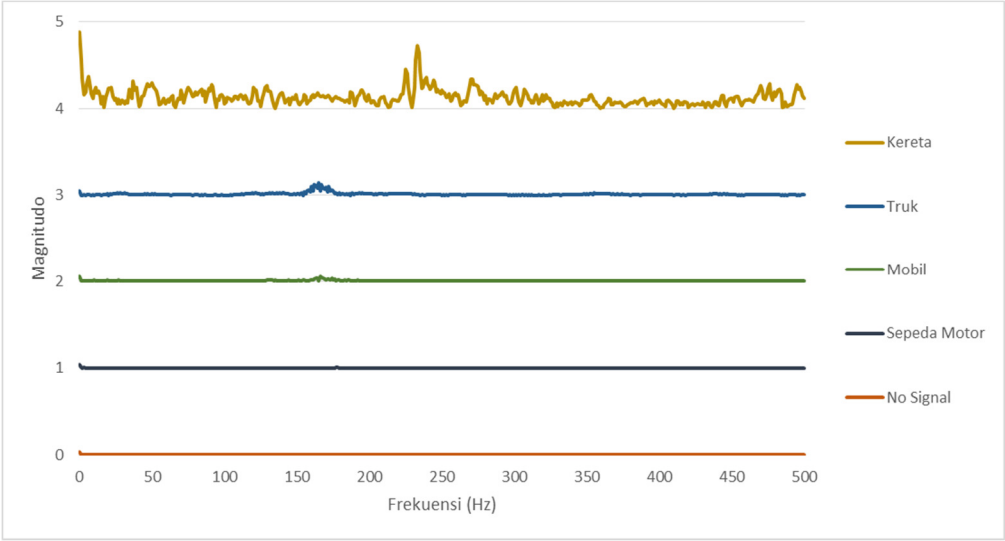


(a) (b)

Gambar 3.8 Sinyal Kalibrasi Sinusoida (a) domain waktu (b) domain frekuensi



Gambar 3.9 Diagram alir Fungsi FFT



Gambar 3.10 Spektrum Frekuensi Ternormalisasi Semua Kondisi

Dalam diagram alir FFT, karena perhitungan fungsi juga dilakukan terhadap 1 frekuensi DC pada 0 MHz, maka jumlah output sebanyak 501 titik frekuensi.

3.6 Normalisasi

Perancangan algoritma Neural Network dimulai dengan normalisasi spektrum data masukan dan data keluaran. Normalisasi data ini dilakukan untuk mempercepat konvergensi saat melakukan learning dan testing, dan juga karena fungsi aktifasi yang digunakan dalam proses identifikasi adalah fungsi aktifasi biner, sehingga data-data tersebut dibawa ke dalam range 0 sampai 1 berdasarkan

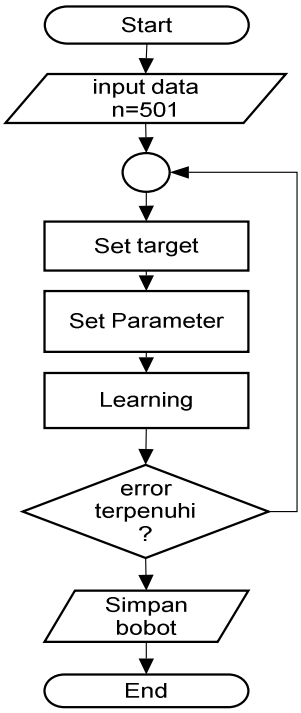
nilai tertinggi dalam data. Spektrum ternormalisasi semua kondisi ditunjukkan dalam Gambar 3.9 di atas.

3.7 Proses Neural Network

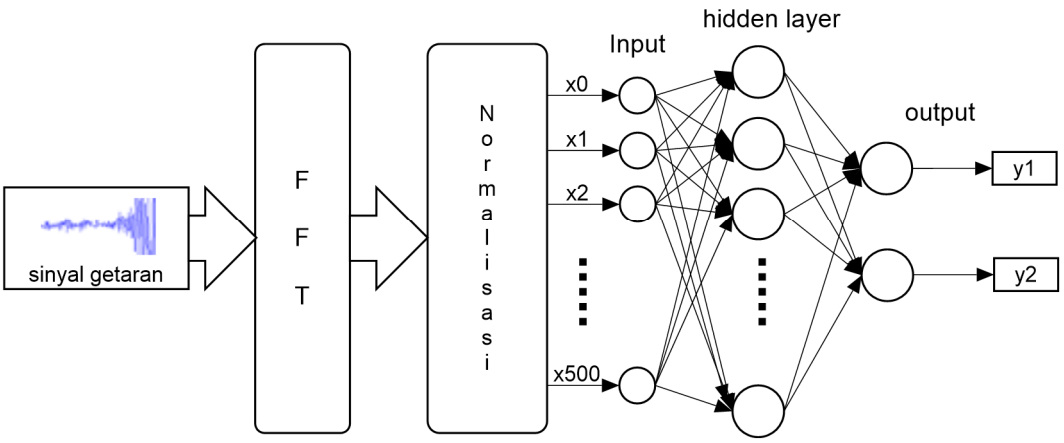
Perancangan neural network pada proses pembelajaran pola frekuensi dapat diilustrasikan dalam diagram alir Gambar 3.11 di bawah.

Input data sejumlah 501 titik untuk menghitung bobot diatur menggunakan parameter $\mu = 0.6$, alpha = 0.3, dan error = 0.0001. Jika error belum terpenuhi, maka perhitungan bobot diulang kembali menggunakan parameter tersebut. Setelah error tercapai, maka nilai-nilai bobot disimpan dalam file teks CSV (*comma separated value*).

Target klasifikasi neural network pada layer output adalah 2 kondisi, yaitu kondisi non-kereta dan kondisi kereta seperti terlihat dalam Gambar 3.12. Jumlah hidden layer yang digunakan adalah satu hidden layer sebanyak 50 neuron. Fungsi aktifasi yang digunakan adalah *binary sigmoid* sehingga nilai target pada proses pembelajaran dibuat dalam bentuk nilai biner.



Gambar 3.11 Diagram Alir Pembelajaran Neural Network

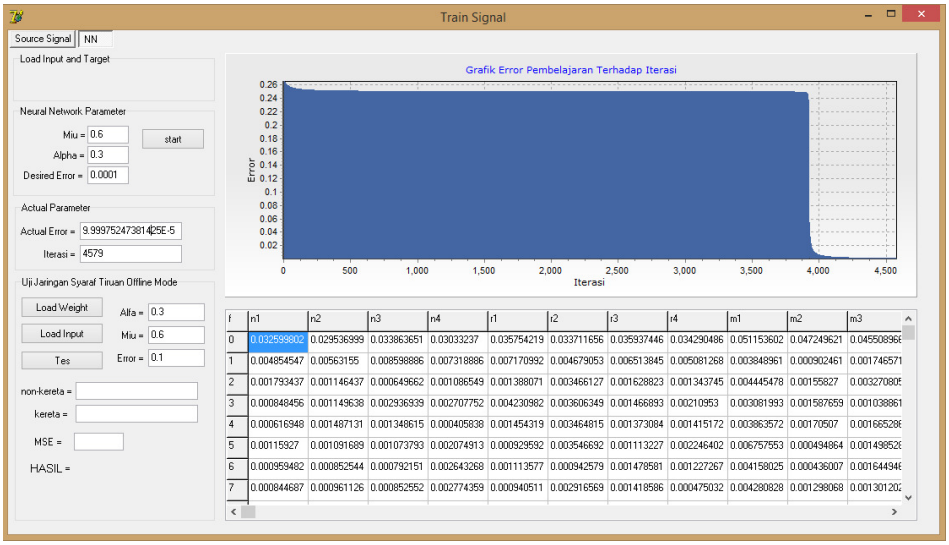


Gambar 3.12 Diagram Identifikasi Neural Network

Tabel 3.2 Matriks target output Neural Network

No.	Kondisi	Kelompok	Matriks
1	No Signal	Non-Kereta	1 0
2	Sepeda Motor		
3	Mobil		
4	Truk		
5	Kereta	Kereta	0 1

Jumlah iterasi proses pembelajaran neural network berlangsung sebanyak 4579 kali untuk mencapai kondisi konvergen. Seperti terlihat dalam Gambar 3.13, pada iterasi ke-4000 nilai error mencapai sekitar 0.02, dan setelah itu mencapai konvergen dengan target error 0.0001.



Gambar 3.13 Program Pembelajaran Neural Network

Tabel 3.3 Jumlah data learning dan data testing.

Kondisi	Data Learning	Data Testing
Non-Kereta	16	4
Kereta	4	4
Total Data	**20**	**8**

Pada tabel 3.3 di atas, jumlah data pembelajaran masing-masing kondisi sebanyak 4 data untuk kondisi no-signal, motor, mobil, dan truk, sehingga keseluruhan sebanyak 16 data learning. Sedangkan data learning kereta api sebanyak 4 data. Dan pada proses pengujian (testing) untuk kondisi non-kereta, masing-masing diujikan 1 data untuk kondisi no-signal, motor, mobil, dan truk (4 data). Dan untuk kondisi kereta api diujikan 4 data testing.

Halaman ini sengaja dikosongkan

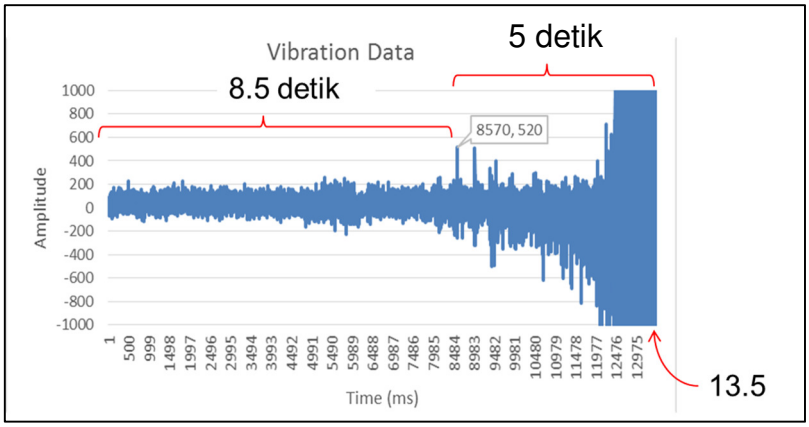
BAB 4
HASIL DAN PEMBAHASAN

4.1 Hasil Deteksi Jarak Kedatangan Kereta

Jarak deteksi kedatangan kereta api yang berhasil dideteksi rata-rata 45 meter dari titik pengukuran. Kondisi pengukuran dilihat dari amplitudo getaran sebesar 400 LSB atau lebih yang terlihat dalam sampel #1 Gambar 4.1 yang muncul pada 8.5 detik (8570 ms).

Pada sampel kereta #1 dalam Tabel 4.1, penghitungan jarak deteksi getaran (xd) adalah waktu deteksi dikalikan kecepatan kereta.

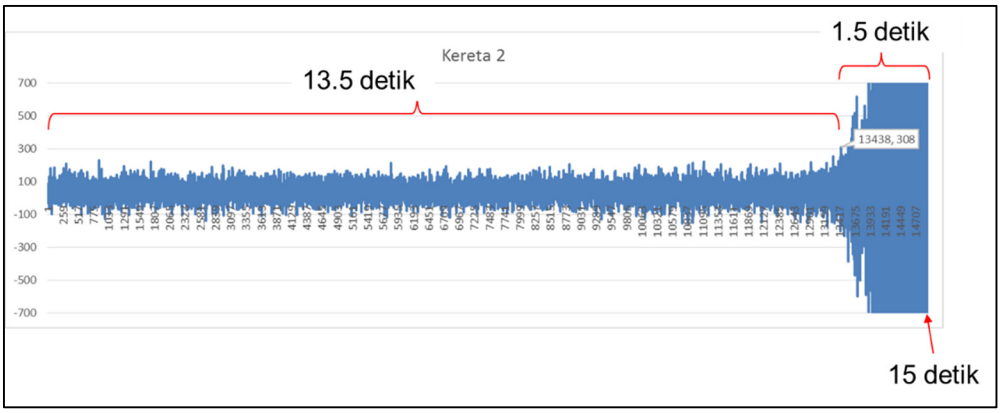
xd = (5) (14.8) = 74 meter



Gambar 4.1. Sinyal Getaran kereta sampel #1.

Tabel 4.1. Deteksi getaran dari jarak 200 meter (warna merah tidak ada gambar)

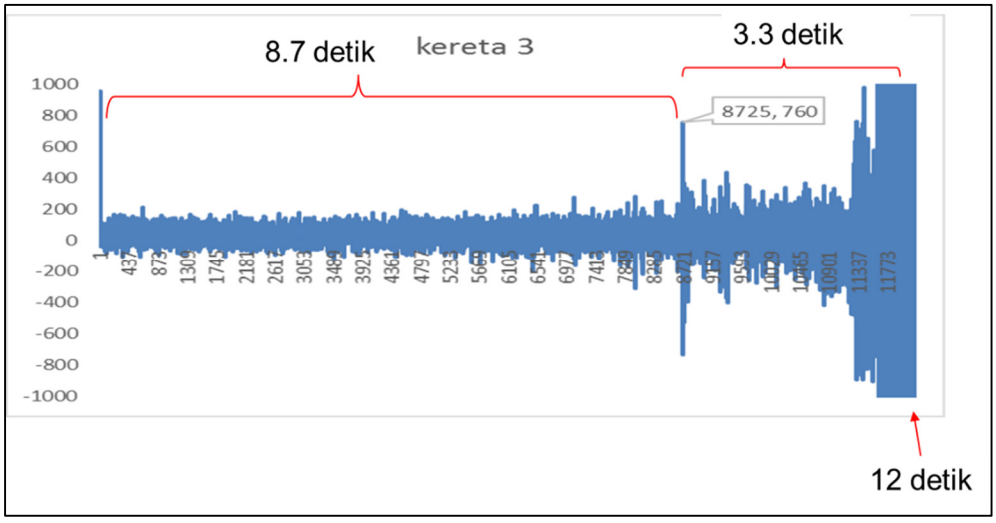
Sampel Kereta	Waktu datang (det)	Kecepatan (m/detik)	Waktu deteksi (det)	Jarak deteksi (m)
#1	13.5	14.8	8.5	74
#2	15	13.3	13	26.6
#3	12	16.7	8.7	55
#4	13.5	14.8	9.2	63
5	13.5	14.8	11	37
Rata-rata				45



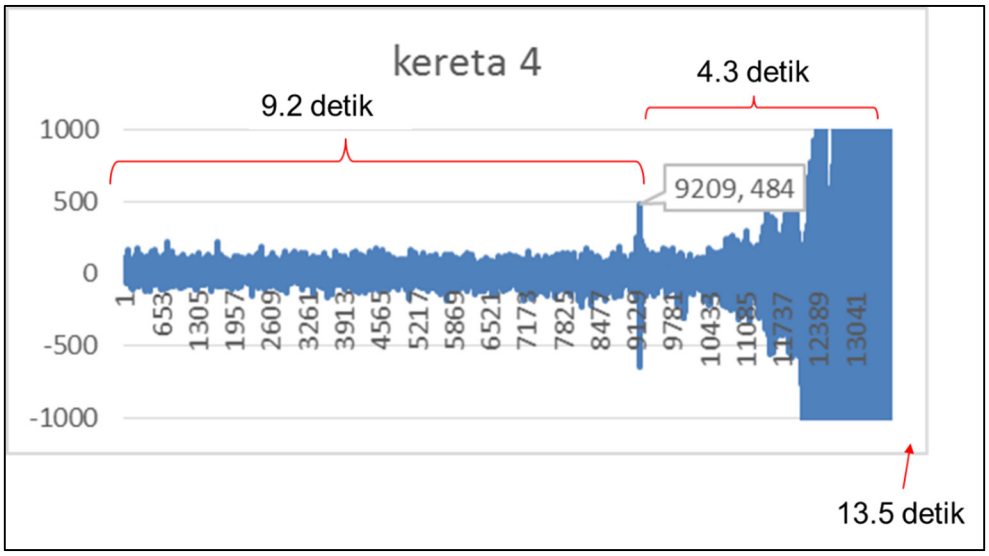
Gambar 4.2 Sinyal Getaran Kereta sampel #2

Pada sampel #2 dalam Gambar 4.2, getaran yang terdeteksi pada 13.5 detik. Kondisi kemunculan sinyal ini lebih lama dibandingkan sampel #1. Hal ini dapat dimungkinkan karena faktor perbedaan jenis kereta, yakni dimensi bobot kereta yang lebih ringan

. Untuk sampel getaran #3 dalam Gambar 4.3 di atas, terdapat lonjakan amplitudo tiba-tiba dari kondisi no-signal yang cukup tinggi padadetik ke 8.7 sebesar 760 LSB di awal deteksi. Hal ini kemungkinan terjadi karena struktur tanah di bawah rel yang tidak rata (naik-turun) sehingga menyebabkan kereta mengalami hentakan.



Gambar 4.3 Sinyal Getaran Kereta sampel #3



Gambar 4.4 Sinyal Getaran Kereta sampel #4

Sampel getaran kereta #4 dalam Gambar 4.4 di atas identik dengan sampel gettaran #3 yang sama-sama terjadi lonjakan amplitudo pada detik ke 9.2 sebesar 484 LSB

Dan pada Gambar 4.5 di bawah menunjukkan spektrum frekuensi masing-masing kondisi, yakni no signal, sepeda motor, mobil, truk, dan kereta api. Dari spektrum ini kemudian dikenali atau diidentifikasi oleh blok Neural Network.

4.2 Hasil Identifikasi Kereta Api dan Non-Kereta Api

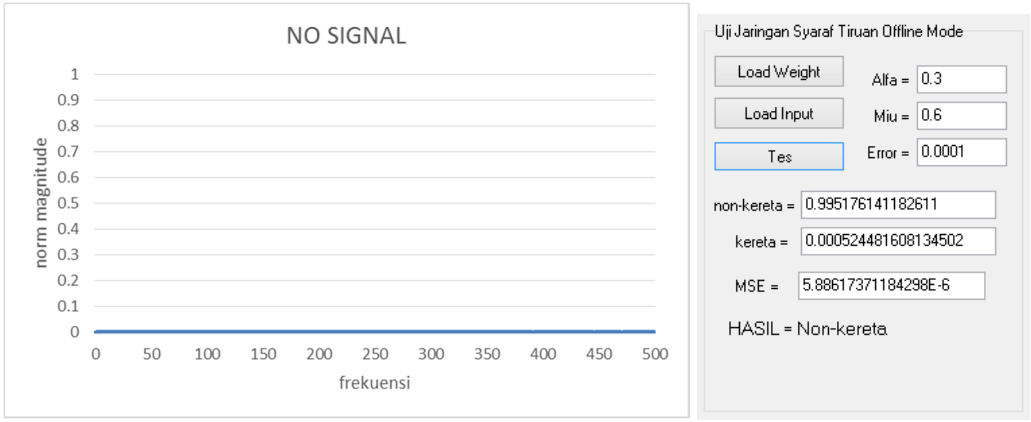
Hasil pengujian proses Neural Network untuk semua kondisi input spektrum dapat dikenali dari semua data pengujiannya sebesar 100%, baik menggunakan target error 0.01, 0.001, maupun 0.0001, seperti terlihat pada Tabel 4.2 di bawah.

Tabel 4.2 Pengujian Identifikasi Neural Network

Kondisi	Pengujian	Hasil		
		0.01	0.001	0.0001
Non-Kereta	1	Non-kereta	Non-kereta	Non-kereta
	2	Non-kereta	Non-kereta	Non-kereta
	3	Non-kereta	Non-kereta	Non-kereta

	4	Non-kereta	Non-kereta	Non-kereta
Kereta	1	Kereta	Kereta	Kereta
	2	Kereta	Kereta	Kereta
	3	Kereta	Kereta	Kereta
	4	Kereta	Kereta	Kereta
Persentase		**100%**	**100%**	**100%**

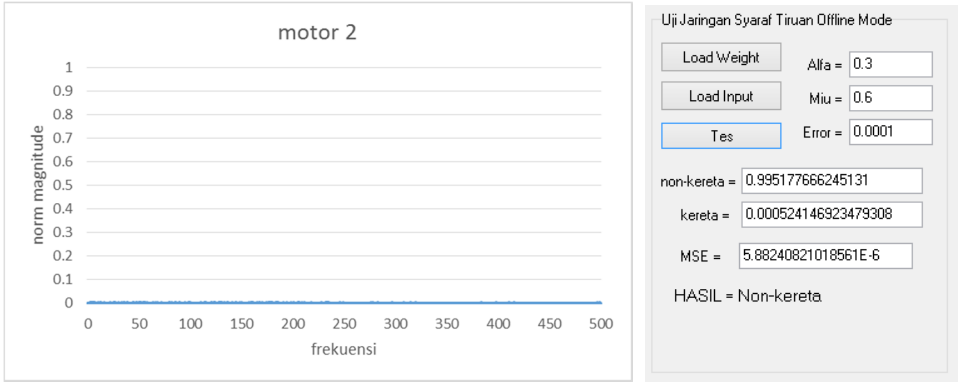
A. Kondisi No Signal (non-kereta)



Gambar 4.5 Hasil Identifikasi No Signal

Pola frekuensi kondisi no-signal berbentuk datar pada semua frekuensi yang menandakan tidak ada frekuensi dominan atau tidak terjadi getaran dari obyek apapun.

B. Kondisi Sepeda Motor (non-kereta)

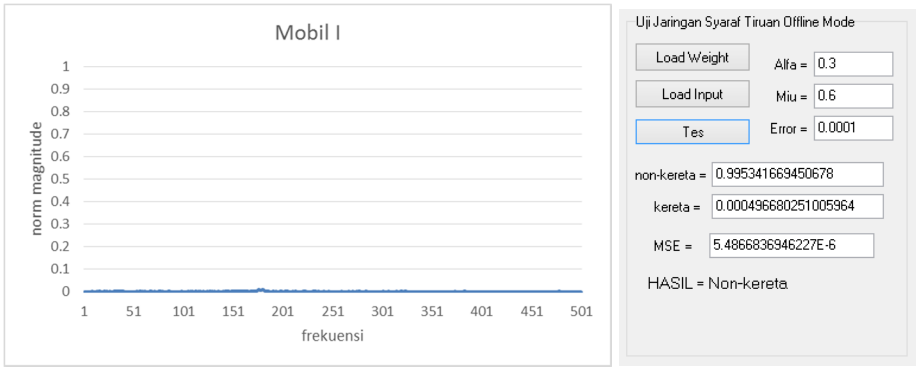


Gambar 4.6. Pola Frekuensi Sepeda Motor testing #2

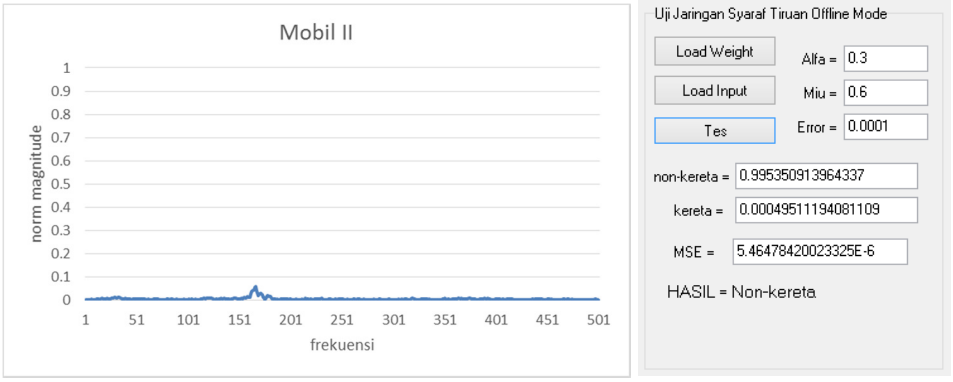
Meskipun terjadi getaran yang disebabkan oleh sepeda motor, namun magnitudo pada pola frekuensinya sangat kecil dan hampir sama dengan kondisi no signal yang datar di semua frekuensi. Hal ini karena telah dilakukan normalisasi berdasarkan magnitudo tertinggi pada semua data learning dan testing.

C. Kondisi Mobil (non-kereta)

Untuk kedua sampel mobil 1 dan mobil 2 dalam Gambar 4.7a dan 4.7b, terdapat perbedaan magnitudo yang tampak pada pola frekuensinya. Hal ini disebabkan karena mobil 2 memiliki berat yang lebih besar atau melaju dengan kecepatan lebih tinggi dibandingan dengan mobil 1 sehingga magnitudonya menjadi lebih tinggi.



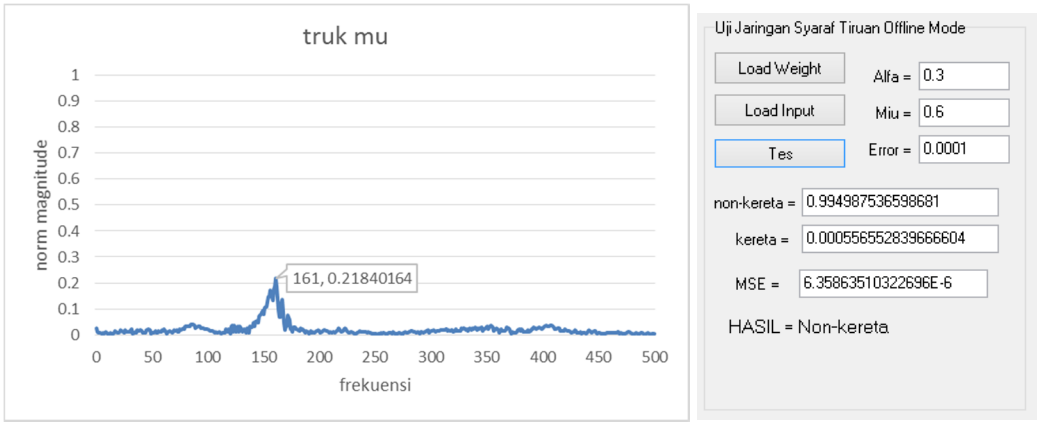
(a)



(b)

Gambar 4.7 Pola Frekuensi Mobil testing #3 (a) Mobil 1 (b) Mobil 2. (frekuensi dominan = 150~200 Hz)

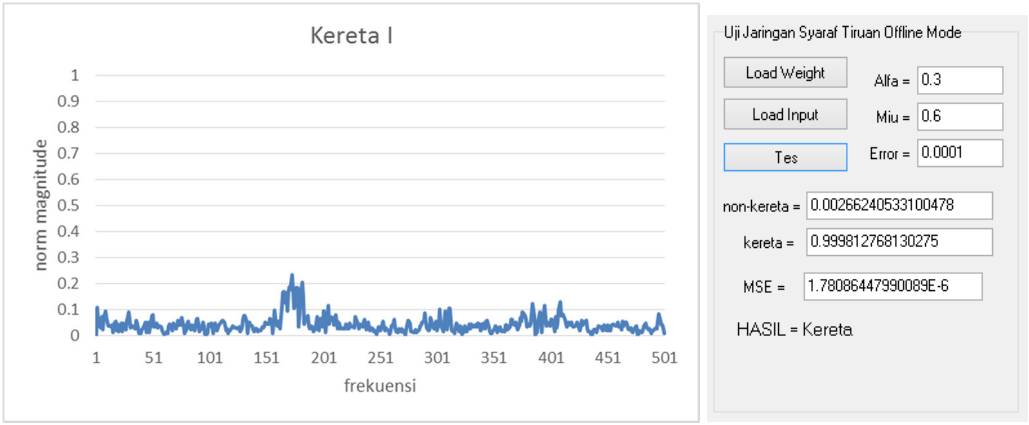
D. Kondisi Truk



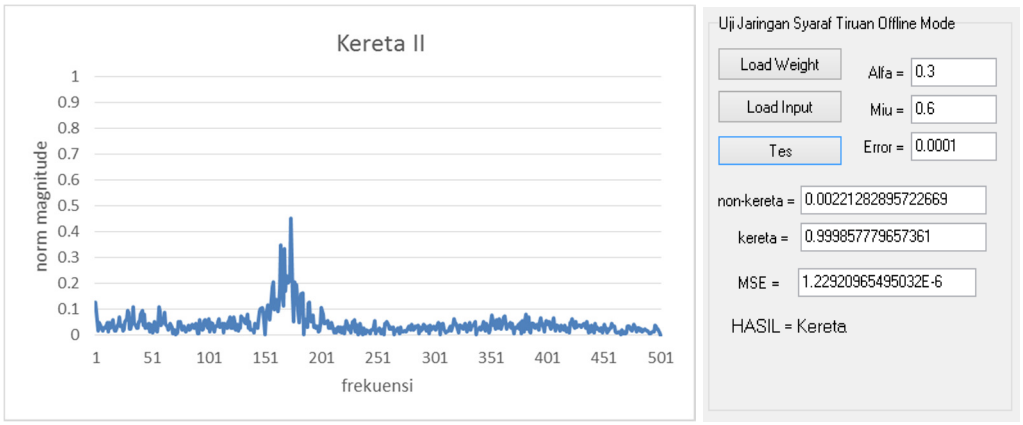
Gambar 4.8 Pola Frekuensi Truk testing #4, frekuensi dominan = 161 Hz

Untuk pola frekuensi truk dalam Gambar 4.8, nampak jelas berbeda magnitudonya dibandingkan pola mobil ataupun sepeda motor dengan ketinggian magnitudo mencapai 0.218. Meskipun demikian, pola ini dapat dikenali dengan baik dalam kelompok non-kereta.

E. Kondisi Kereta



(a)



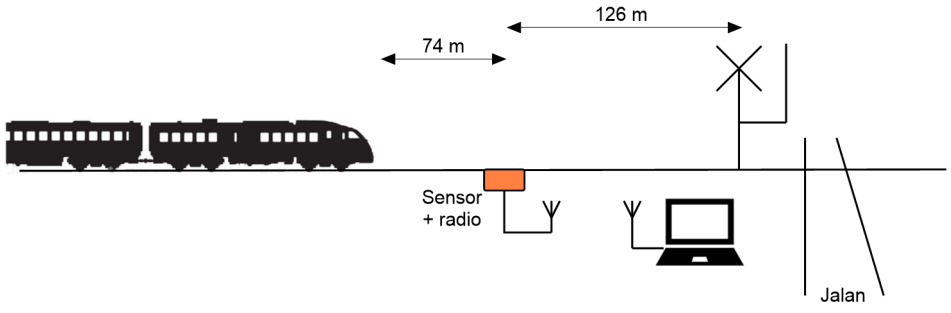
(b)

Gambar 4.9 Pola Frekuensi Kereta testing #1, #2 (a) Kereta 1 (b) Kereta 2

Dari bentuk spektrum kereta yang dihasilkan, terlihat bahwa di setiap titik terdapat riakan frekuensi. Tidak seperti pada spektrum non-kereta yang bentuk spektrumnya lebih datar di luar frekuensi dominannya. Hal ini yang membuat blok Neural Network dapat dengan mudah membedakan antara sinyal getaran kereta dan sinyal getaran non-kereta.

4.3 Modifikasi Perancangan Sistem untuk Pengembangan

Meskipun secara keseluruhan pola frekuensi dapat diidentifikasi, namun hasil penelitian menunjukkan deteksi kereta terjauh pada posisi 74 meter dari perlintasan Oleh karena itu, sebaiknya dilakukan pergeseran letak sensor sejauh 126 meter dari perlintasan seperti dalam Gambar 4.10 agar sistem peringatan dapat difungsikan secara maksimal.



Gambar 4.10 Sistem Peringatan menggunakan Radio

Halaman ini sengaja dikosongkan

BAB 5
KESIMPULAN

Pada penelitian ini telah dibuat suatu sistem peringatan kedatangan kereta api, sistem ini terdiri dari MEMS Accelerometer, Arduino Uno, Algoritma Fast Fourier Transform, dan Neural Network. Sensor accelerometer digunakan untuk membaca getaran pada rel kereta api. Pada sistem ini menggunakan sensor accelerometer MPU6050 dan 16-bit Analog to Digital Converter sehingga dapat menghasilkan rentang pengukuran sebesar 16384 LSB/g (-32768 ~ 32768). Kekuatan getaran yang terukur menjadi data untuk memperkirakan posisi kedatangan kereta api. Sinyal getaran dalam domain waktu tersebut dikonversi menjadi spektrum frekuensi menggunakan Fast Fourier Transform sebanyak 501 komponen frekuensi (0 ~ 500 Hz). Algoritma Neural Network 3-lapis digunakan untuk mengenali pola frekuensi getaran rel yang disebabkan oleh kereta api yang dapat dibedakan dari sepeda motor, mobil, dan truk yang melewati perlintasan. Mikrokontroler Arduino Uno digunakan untuk membaca data sensor dan mengirimkannya ke komputer. Hasil pengujian menunjukkan bahwa getaran kereta api dapat dideteksi pada jarak terjauh hingga 74 meter dan dapat mengenali pola getaran kereta api terhadap yang lainnya dengan tingkat keberhasilan sebesar 100%. Untuk pengembangan penelitian ini, penempatan sensor dilakukan sejauh 126 meter dari perlintasan. Kemudian data hasil pembacaan sensor dikirimkan melalui komunikasi radio.

Halaman ini sengaja dikosongkan

DAFTAR PUSAKA

Angrisani, L., Grillo, D., Moriello, R.S.L., Filo, G., (2010), "Automatic Detection of Train Arrival Through An Accelerometer", *Instrumentation and Measurement Technology Conference (I2MTC)*, IEEE, Austin, hal. 898-902.

Diniz, P.S.R., da Silva, E.A.B., Netto, S.L., (2010), *Digital Signal Processing*, 2nd Edition, Cambridge University Press, New York.

InvenSense, (2013*), "*MPU6050 Product Specification", InvenSense Inc, USA.

Jiang, B. and Wang, Z., (2014), "Railway Vibration Monitoring System based on ARM and Acceleration". *IEEE 9th Conference on Industrial Electronics and Applications (ICIEA)*, IEEE, Hangzhou, hal. 998-1004.

Lungu, G. (2011). "Spectral Analysis a Fourier transform tutorial", http://www.excelunusual.com/spectral-analysis-a-fourier-transform-tutorial-part-5/. Diakses tanggal 13-11-2017.

PR KAI Divre 1. "Bahaya Dan Sanksi Menerobos Pintu Perlintasan". Pers PT. KAI, https://kai.id/information/full_news/713-bahaya-dan-sanksi-menerobos-pintu-perlintasan. diakses tgl. 3-12-2017.

Rivai, M., Tasripan, Sumandri. (2007). "Pengenalan Pola Sinyal Suara Kerusakan Motor Listrik menggunakan Neural Network". *Journal of Electrical and Electronics Engineering.* JAVA Vol. 5, No.2, ISSN 1412-8306.

Santos, J., Hempel, M., Sharif, H., (2013), "Sensing Techniques and Detection Methods for Train Approach Detection", *IEEE 78th Vehicular Technology Conference (VTC Fall)*, IEEE, Las Vegas, hal. 1-5.

Santoso, B.D., (2014), *Gambaran Sistem Persinyalan Westrace* http://bayu12tav1.blogspot.com, diakses Agustus 10, 2015.

Wang, C., Xiao, Q., Liang, H., Chen, X., Cai, X., Liu, Y., (2006), "On-Line Vibration Source Detection of Running Trains Based on Acceleration Measurement". *IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, Beijing, hal. 4411-4416.

wikimapia.org, (2013), Gardu Petugas Perlintasan Jalan Kereta Api no. 162, http://wikimapia.org/27842567/id/Gardu-PJL-162-Braga-Bandung. diakses Agustus 10, 2015.

LAMPIRAN

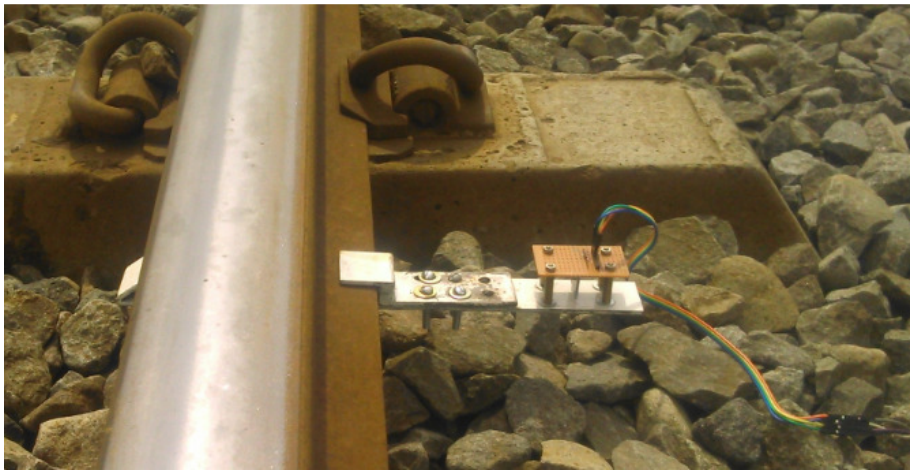
Foto Pemasangan Alat

Tampak Perlintasan

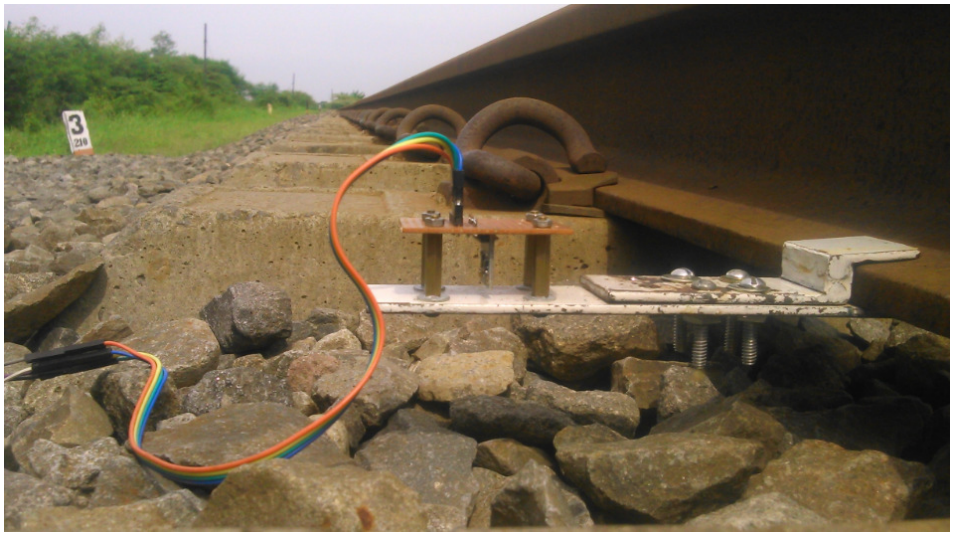
Arah Kedatangan Kereta

Tampak Atas, Modul Sensor pada Braket

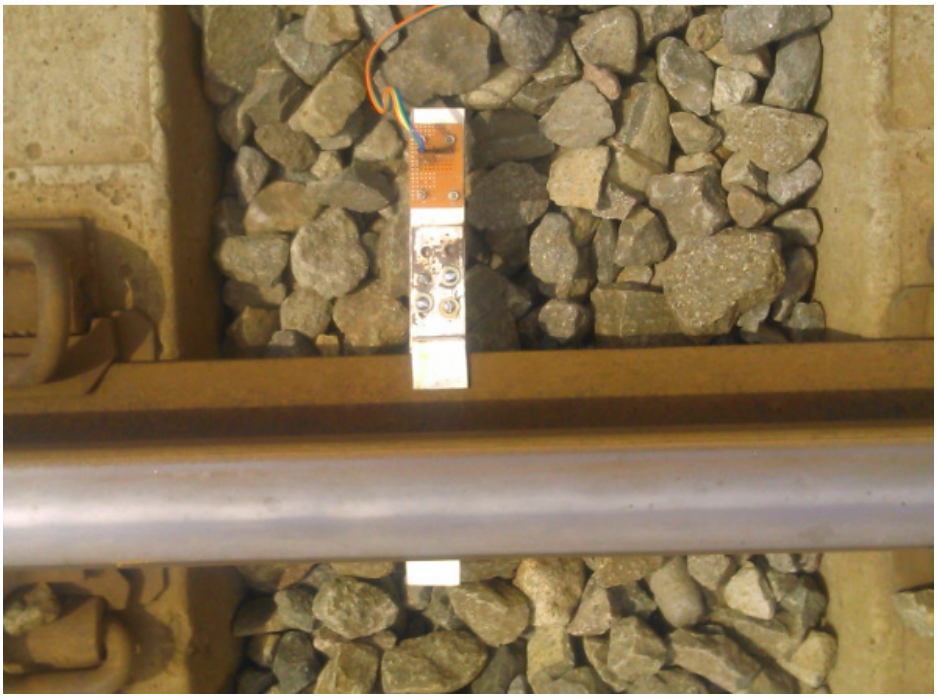
Tampak Sisi dalam Braket



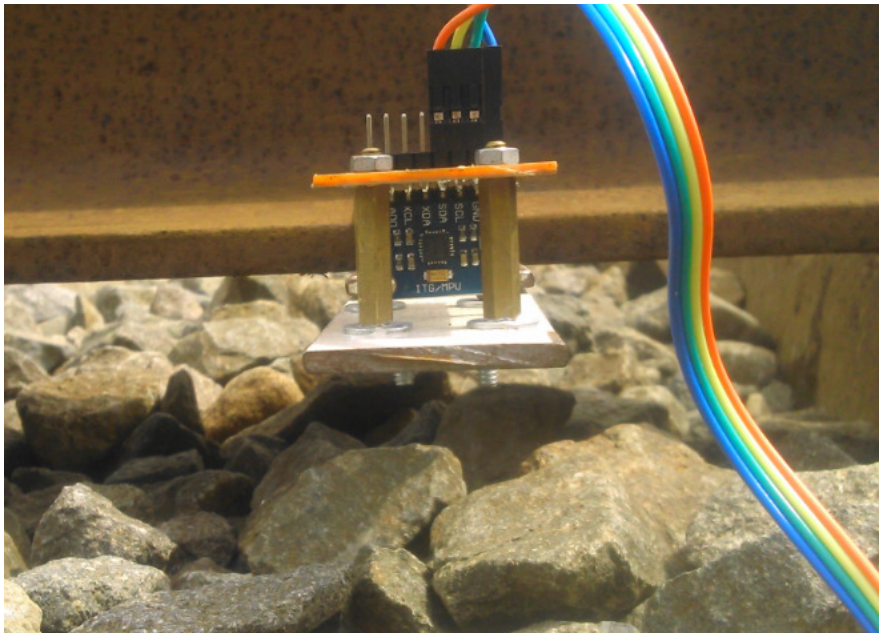
Tampak Samping



Tampak Atas

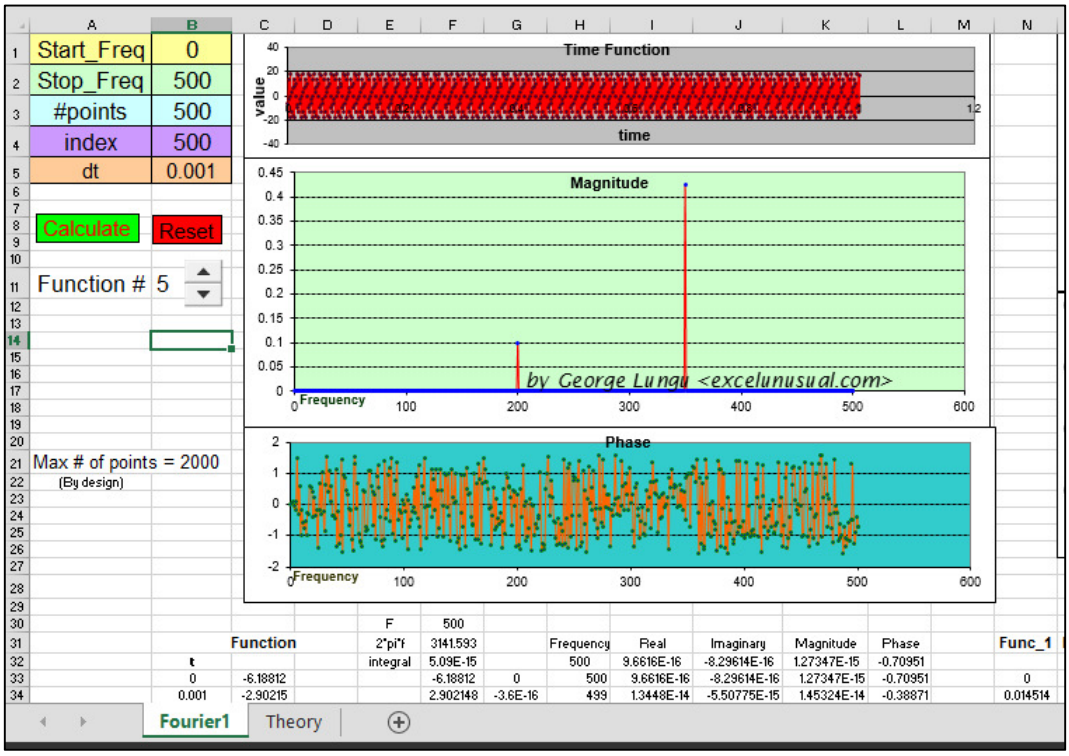


Tampak dari Sisi Luar

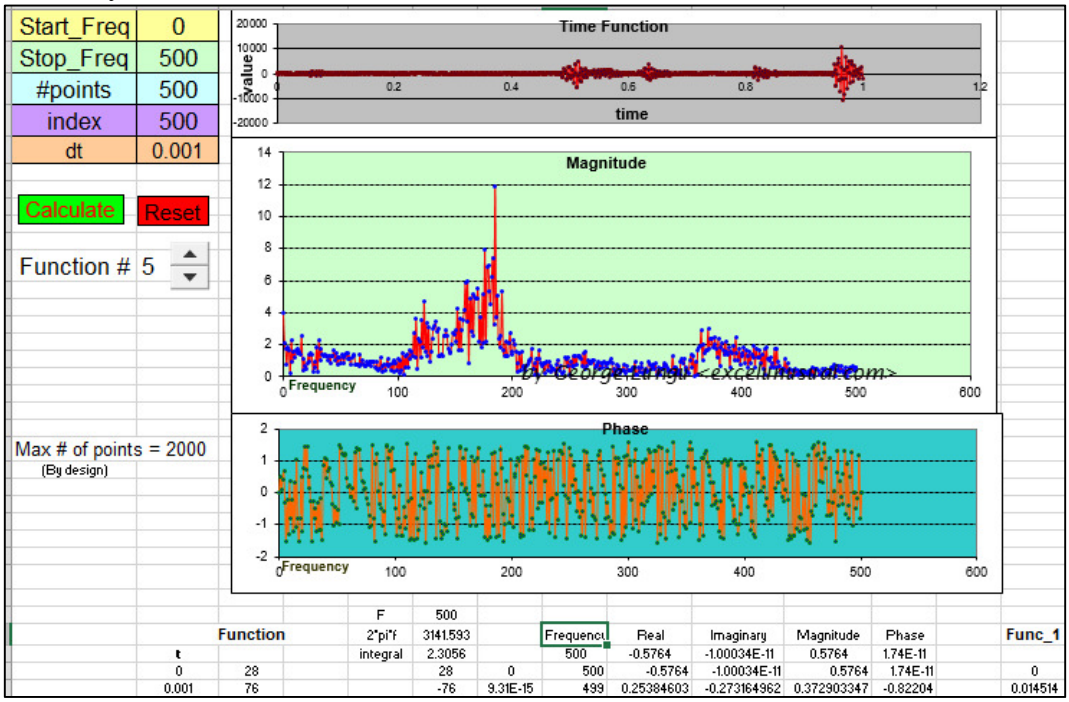


Proses FFT menggunakan MS Excel

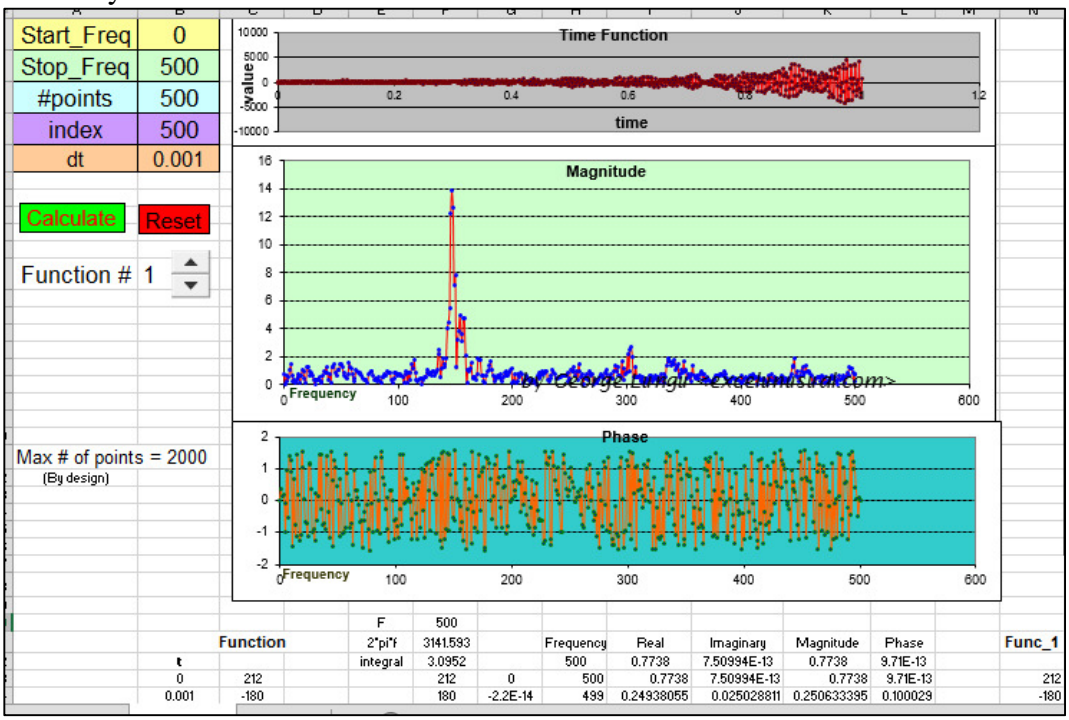
FFT Kalibrasi Sinusoida



FFT Sinyal Truk

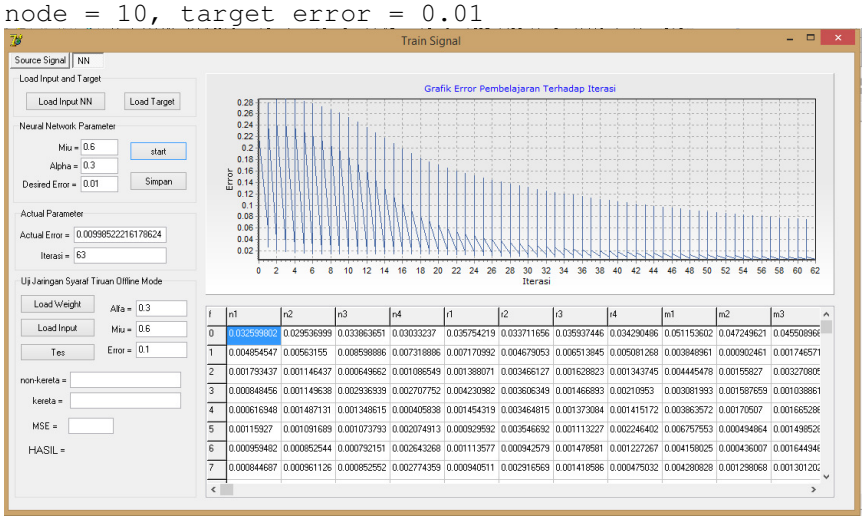


FFT Sinyal Kereta

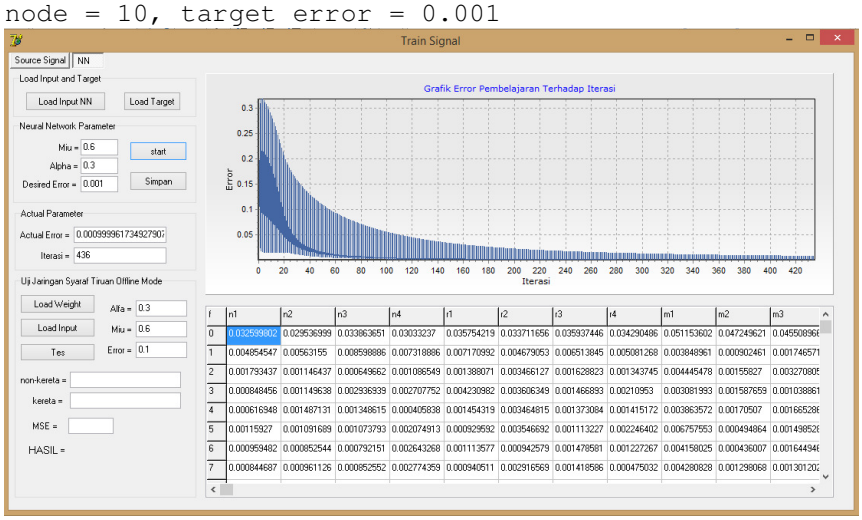


Proses Pembelajaran / Learning Neural Network

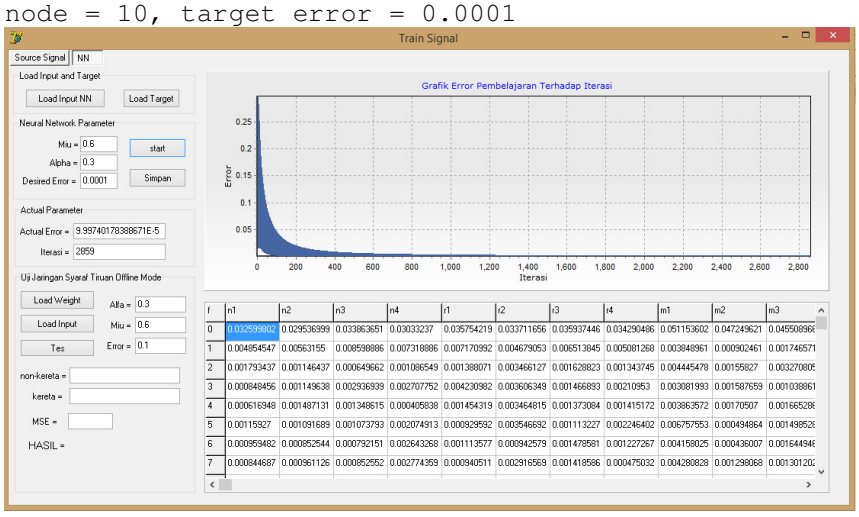
```
node = 10, target error = 0.01
```



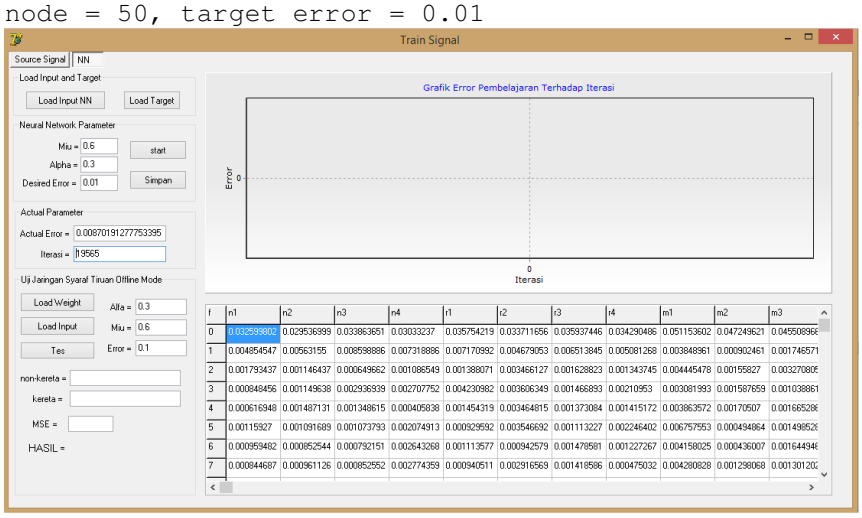
node = 10, target error = 0.001



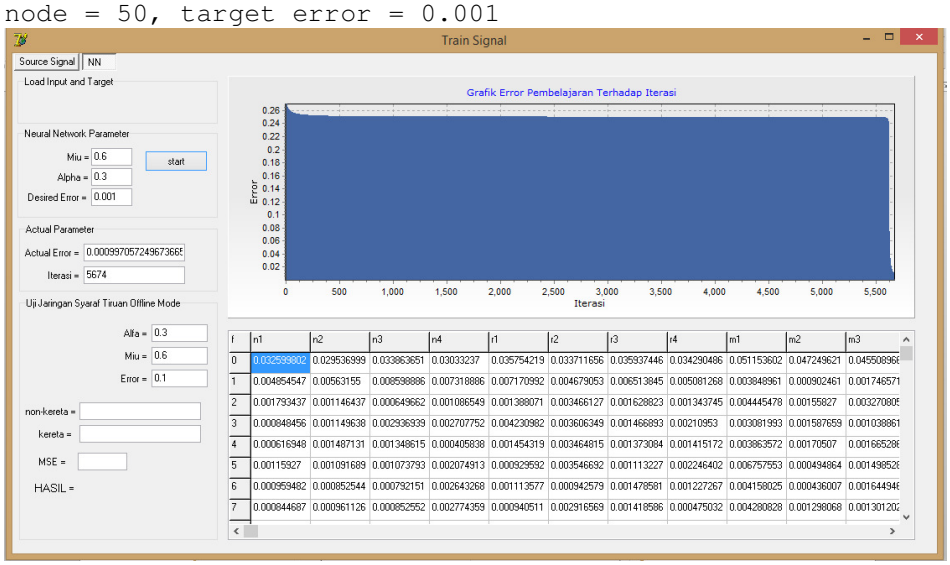
node = 10, target error = 0.0001



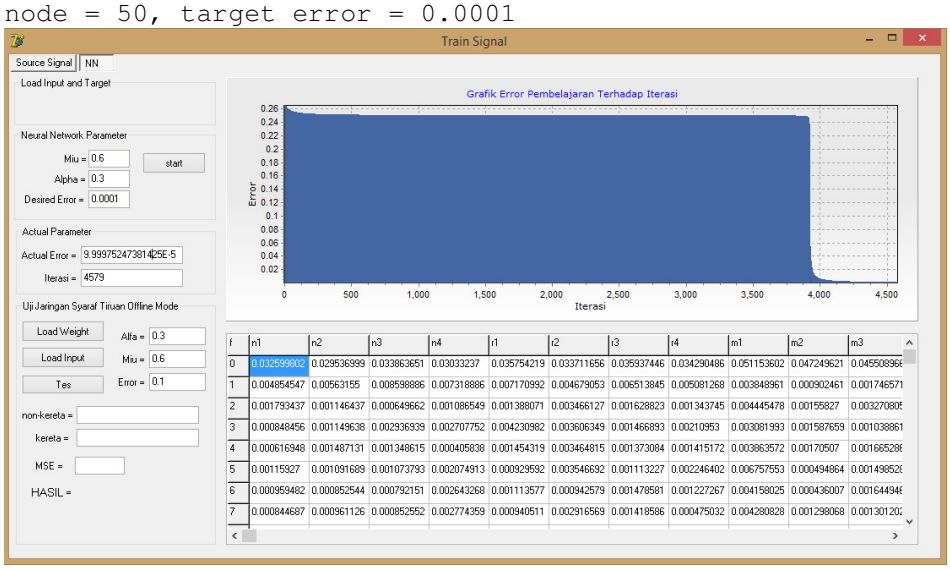
node = 50, target error = 0.01



node = 50, target error = 0.001

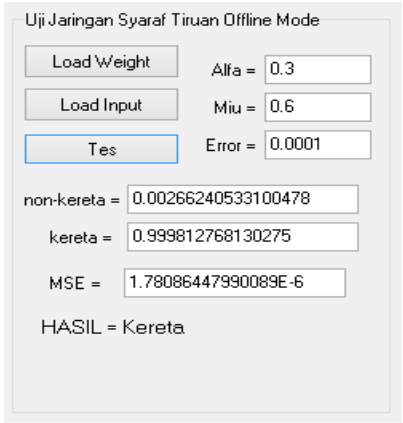


node = 50, target error = 0.0001

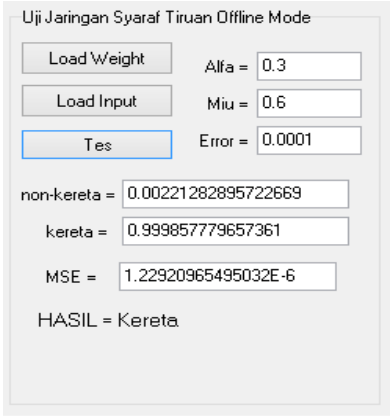


Proses Pengujian / Testing Neural Network

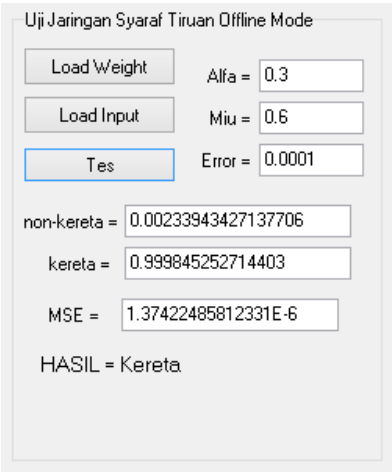
Kereta 1



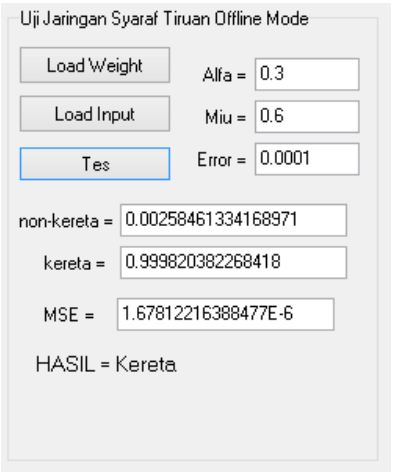
Kereta 2



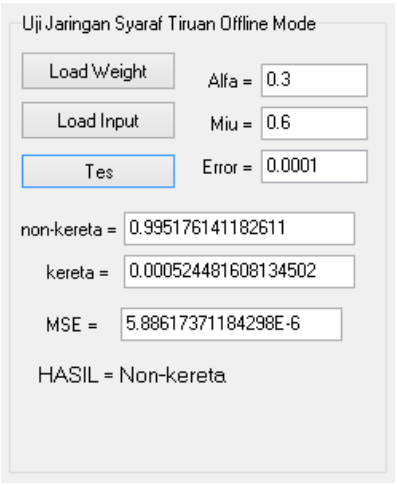
Kereta 3



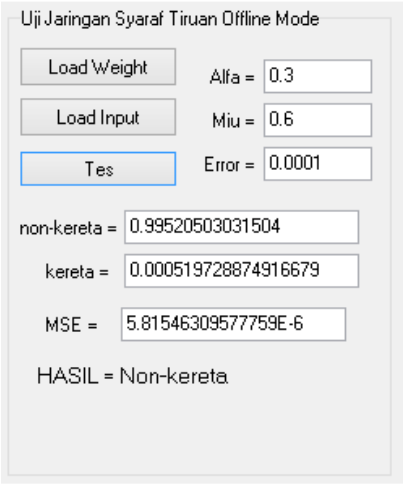
Kereta 4



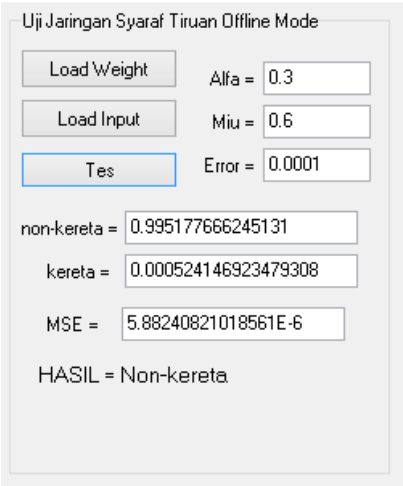
No Signal 1



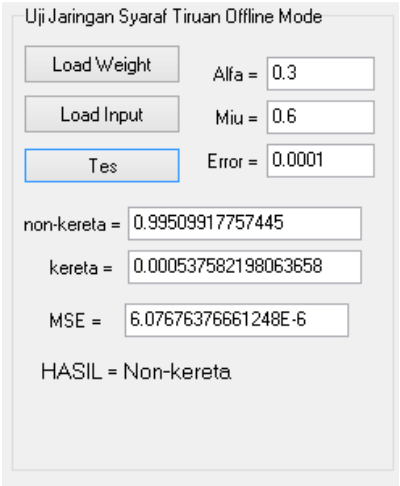
No Signal 2



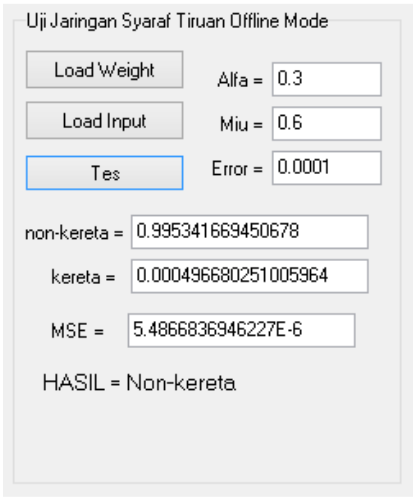
Sepeda Motor 1



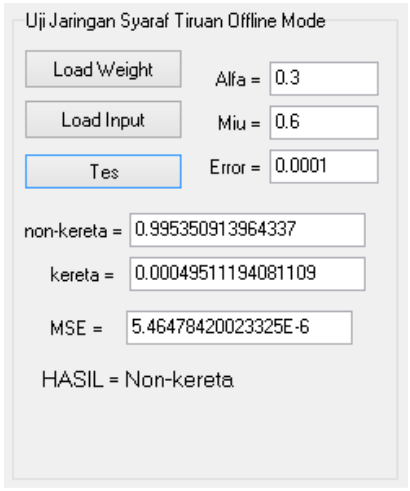
Sepeda Motor 2



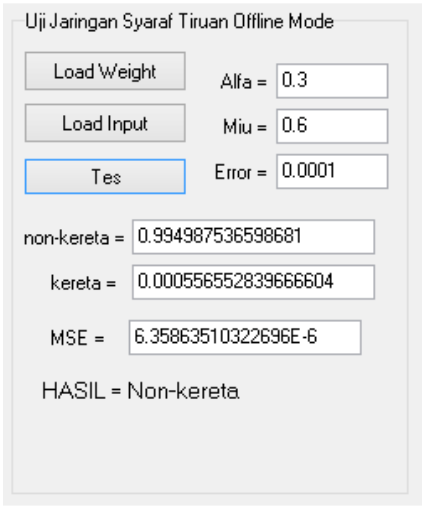
Mobil 1



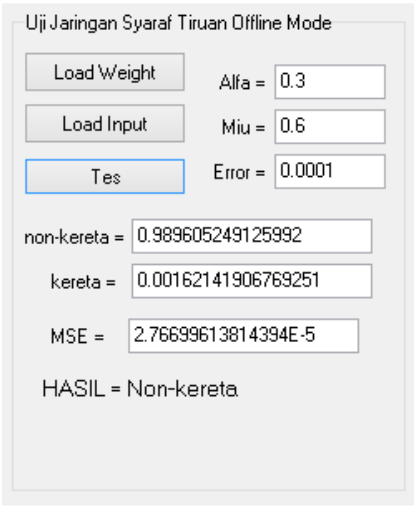
Mobil 2



Truk 1



Truk 2



```
unit uType;
interface
uses
SysUtils;
type
AData = array of double;
AAData = array of array of Double; //[row,col]
PAData = array of AData;
PAAData = array of AAData;
AInt = array of Integer;
psample = ^smallint ;
implementation
end.

unit uMain;
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms,
Dialogs, StdCtrls, TeEngine, Series,ExtCtrls, TeeProcs, Chart,Grids,
ComCtrls,Math, XPMan,
TeeFunci, uFilter, uFileData, SLCommonFilter,
ALCommonFilter, ALBasicGenericFilter, ALGenericFilter, LPComponent,
ALAudioIn, ALCommonLogger, ALWaveLogger,uType;
type
TFSpeech = class(TForm)
SaveDialog1: TSaveDialog;
PageControl1: TPageControl;
TabSheet6: TTabSheet;
cSignal: TChart;
Series1: TFastLineSeries;
cFFT: TChart;
NN: TTabSheet;
groupbox2: TGroupBox;
btn1: TButton;
btn2: TButton;
groupbox3: TGroupBox;
lbl1: TLabel;
lbl2: TLabel;
lbl3: TLabel;
Edit9: TEdit;
Edit10: TEdit;
Edit11: TEdit;
btn3: TButton;
btn4: TButton;
groupbox4: TGroupBox;
lbl4: TLabel;
lbl5: TLabel;
Edit12: TEdit;
Edit17: TEdit;
groupbox6: TGroupBox;
lbl6: TLabel;
lbl7: TLabel;
lbl8: TLabel;
lbl9: TLabel;
lbl10: TLabel;
lbl11: TLabel;
Label34: TLabel;
btn5: TButton;
btn6: TButton;
btn7: TButton;
```

```
Edit18: TEdit;
Edit19: TEdit;
Edit22: TEdit;
Edit23: TEdit;
Edit24: TEdit;
Edit25: TEdit;
chart1: TChart;
Series30: TLineSeries;
sg1Grid1: TStringGrid;
timer3: TTimer;
dlgOpen1: TOpenDialog;
Timer1: TTimer;
Button1: TButton;
ALAudioIn1: TALAudioIn;
ALGenericFilter1: TALGenericFilter;
Series2: TLineSeries;
Series3: TLineSeries;
Button2: TButton;
Edit2: TEdit;
Edit3: TEdit;
Label1: TLabel;
Label2: TLabel;
Chart2: TChart;
Series4: TLineSeries;
Chart3: TChart;
Series5: TLineSeries;
ALWaveLogger1: TALWaveLogger;
Chart4: TChart;
Series6: TFastLineSeries;
Chart5: TChart;
Series7: TFastLineSeries;
cbIden: TCheckBox;
lResultIden: TLabel;
Label3: TLabel;
procedure btn1Click(Sender: TObject);
procedure btn2Click(Sender: TObject);
procedure btn3Click(Sender: TObject);
procedure btn4Click(Sender: TObject);
procedure btn5Click(Sender: TObject);
procedure btn6Click(Sender: TObject);
procedure btn7Click(Sender: TObject);
procedure timer3Timer(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure Timer1Timer(Sender: TObject);
procedure ALGenericFilter1ProcessData(Sender: TObject;
InBuffer: IALAudioBuffer; var OutBuffer: IALAudioBuffer;
var SendOutputData: Boolean);
procedure dft(a: array of double);
procedure Button2Click(Sender: TObject);
procedure TabSheet6ContextPopup(Sender: TObject; MousePos: TPoint;
var Handled: Boolean);
procedure FormCreate(Sender: TObject);
//procedure FormCreate(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
procedure random_weight();
procedure maju();
procedure mundur();
procedure update_weight();
procedure loadWeight;
end;
var
```

```
FSpeech: TFSpeech;
out_count,amp,segment,orde,frame,nfft,pitchP: integer;
xm : PAData;
em,xmnew,fx,fxnew:AData;
max1,maxvd,mf1,mf2,threshold:Double;
sig,sigf: array[0..1024] of double;
get_trshld:integer;
xt: array[0..1024] of double;
fourier,fourier_frek: array[0..1024] of double;
ftr: array[0..1024,0..2] of double;
WijN: array[0..1024,0..1024,0..2] of double;
lala: integer;
vad_power: array[0..100] of double;
vad_frekuensi: array [0..100] of double;
vad_frekuensi_th,vad_power_th: double;
//Neural Network
input_NN,desired:AAData;
iterasi,z,patern: integer;
n_input,n_hidden,n_output:integer;
alfa,miu,galatmin,galat_total: double;
erkum: AData;
w1,w2,bias1,bias2:AAData;
node1,out1,turunan1,error1:AData;
node2,out2,turunan2,error2: AData;
nnok:boolean;
setpoint:Double;
pos_MSE: integer;
fhasil:Adata;
const
dbase_pengenalan: array[0..1,0..1] of integer= ((1,0),(0,1));
dbase_string: array[0..1] of string= ('Non-kereta','Kereta');
N: integer=1024;
sampling: integer=44100;
implementation
{$R *.dfm}
//=================fungsi W ijn real============================
function wijn_re(pola_frek,time_step,jumlah_data: integer): double;
begin
result:= cos(2*pi*time_step*pola_frek/jumlah_data);
end;
//==================fungsi W ijn imajiner=====================
function wijn_im(pola_frek,time_step,jumlah_data: integer): double;
begin
result:= -sin(2*pi*time_step*pola_frek/jumlah_data);
end;
//vad power
function vad_pow(iterasi: integer): double;
var
i:integer;
tmp: double;
begin
i:= 0;
tmp:=0;
while i< N do
begin
tmp:= tmp+ sqr(xt[i]);
i:=i+1;
end;
result:= tmp;
end;
function vad_frek(iterasi: integer): double;
var
i: integer;
max_val,frek: double;
```

```
begin
max_val:=0;
i:=0;
frek:=0;
while i< (N/2) do
begin
if fourier[i]>max_val then
begin
max_val:= fourier[i];
frek:= fourier_frek[i];
end;
i:=i+1;
end;
result:= frek;
end;
//================Neural network============================
function randomrange(low,high:double):double;
var temp:integer;
begin
temp:=Random(1000);
while temp=0 do
temp:=Random(1000);
result:=temp*(high-low)*0.001+low;
end;
//==================random weight=======================
procedure TFSpeech.random_weight();
var
i,j: integer;
begin
for i:= 0 to n_input-1 do
begin
for j:= 0 to n_hidden-1 do
begin
w1[i,j]:= randomrange(0,1);
bias1[0,j]:= randomrange(0,1);
end;
end;
for i:= 0 to n_hidden-1 do
begin
for j:= 0 to n_output-1 do
begin
w2[i,j]:= randomrange(0,1);
bias2[0,j]:= randomrange(0,1);
end;
end;
end;
//==================propagasi maju============================
procedure TFSpeech.maju();
var
i,j: integer;
begin
//propagasi maju untuk layer 1
for i:= 0 to n_hidden-1 do
begin
node1[i]:= 0;
for j:= 0 to n_input-1 do
begin
node1[i]:= node1[i]+(input_nn[j,z]*w1[j,i]);
end;
node1[i]:=node1[i]+bias1[0,i];
out1[i]:=1/(1+exp(-1*alfa*node1[i]));
turunan1[i]:= alfa*(out1[i])*(1-out1[i]);
end;
//propagasi maju untuk output layer (layer 2)
```

```
for i:= 0 to n_output-1 do
begin
node2[i]:= 0;
for j:= 1 to n_hidden-1 do
begin
node2[i]:= node2[i]+(out1[j]*w2[j,i]);
end;
node2[i]:=node2[i]+bias2[0,i];
out2[i]:=1/(1+exp(-1*alfa*node2[i]));
turunan2[i]:= alfa*(out2[i])*(1-out2[i]);
end;
end;
//==================propagasi mundur================================
procedure TFSpeech.mundur();
var
i,j: integer;
begin
erkum[z]:= 0;
for i:= 0 to n_output-1 do //error output (layer 3)
begin
erkum[z]:= erkum[z]+(sqr(desired[i,z]-out2[i]));
error2[i]:= (desired[i,z]-out2[i])*turunan2[i];
end;
erkum[z]:= (0.25)*erkum[z];
for i:=0 to n_hidden-1 do
begin
error1[i]:= 0;
for j:= 0 to n_output-1 do
begin
error1[i]:= error1[i]+(error2[j]*w2[i,j]);
end;
error1[i]:= error1[i]*turunan1[i]; //error layer 1
end;
end;
//================update weight=================================
procedure TFSpeech.update_weight();
var
i,j : integer;
begin
for i:= 0 to n_input-1 do
begin
for j:= 0 to n_hidden-1 do
begin
w1[i,j]:= w1[i,j]+(miu*error1[j]*input_nn[i,z]); //update weight layer 1
bias1[0,j]:=bias1[0,j]+(miu*error1[j]);
end;
end;
for i:= 0 to n_hidden-1 do
begin
for j:= 0 to n_output-1 do
begin
w2[i,j]:= w2[i,j]+(miu*error2[j]*out1[i]); //update weight layer 2
bias2[0,j]:=bias2[0,j]+(miu*error2[j]);
end;
end;
end;
//============================================================
procedure TFSpeech.loadWeight;
var filename:string;
begin
filename:=ExtractFilePath(Application.ExeName)+'NN\';
w1:=nil;
w2:=nil;
bias1:=nil;
```

```
bias2:=nil;
CSVToArray(w1,filename+'nn_w1.csv');
CSVToArray(w2,filename+'nn_w2.csv');
CSVToArray(bias1,filename+'nn_bias1.csv');
CSVToArray(bias2,filename+'nn_bias2.csv');
end;
procedure TFSpeech.btn1Click(Sender: TObject);
begin
ClearStringGrid(sg1Grid1);
dlgOpen1.InitialDir:=ExtractFilePath(Application.ExeName)+'Data\NN\';
if dlgOpen1.Execute then
begin
CSVToSG(sg1Grid1,dlgOpen1.FileName);
SGToArray(sg1Grid1,input_NN);
SetLength(erkum,Length(input_NN[0]));
end;
end;
procedure TFSpeech.btn2Click(Sender: TObject);
begin
dlgOpen1.InitialDir:=ExtractFilePath(Application.ExeName)+'data\NN\';
if dlgOpen1.Execute then
begin
CSVToArrayH(desired,dlgOpen1.FileName);
end;
end;
procedure TFSpeech.btn3Click(Sender: TObject);
begin
if btn3.Caption= 'start' then
begin
btn3.Caption:= 'stop';
n_input:=Length(input_NN);
n_output:=Length(desired);
n_hidden:=50;
SetLength(w1,n_input,n_hidden);
SetLength(w2,n_hidden,n_output);
SetLength(node1,n_hidden);
SetLength(out1,n_hidden);
SetLength(turunan1,n_hidden);
SetLength(error1,n_hidden);
SetLength(bias1,1,n_hidden);
SetLength(node2,n_output);
SetLength(out2,n_output);
SetLength(turunan2,n_output);
SetLength(error2,n_output);
SetLength(bias2,1,n_output);
random_weight();
Series30.Clear;
galat_total:= 1;
iterasi:= 0;
miu:= strtofloat(Edit9.Text);
alfa:= strtofloat(Edit10.Text);
galatmin:= strtofloat(edit11.Text);
timer3.Interval:= 1;
timer3.Enabled:= true;
end
else if btn3.Caption= 'stop' then
begin
btn3.Caption:= 'start';
timer3.Enabled:= false;
end;
end;
procedure TFSpeech.btn4Click(Sender: TObject);
var filename:string;
begin
```

```
SaveDialog1.InitialDir:=ExtractFilePath(Application.ExeName)+'Data\NN\';
if SaveDialog1.Execute then
begin
filename:=SaveDialog1.FileName;
Delete(filename,Pos('.csv',filename),Length(filename));
ArrayToCSV(w1,filename+'_w1.csv');
ArrayToCSV(w2,filename+'_w2.csv');
ArrayToCSV(bias1,filename+'_bias1.csv');
ArrayToCSV(bias2,filename+'_bias2.csv');
end;
end;
procedure TFSpeech.btn5Click(Sender: TObject);
var filename:string;
begin
dlgOpen1.InitialDir:=ExtractFilePath(Application.ExeName)+'Data\NN\';
if dlgOpen1.Execute then
begin
filename:=dlgOpen1.FileName;
Delete(filename,Pos('_',filename),Length(filename));
CSVToArray(w1,filename+'_w1.csv');
CSVToArray(w2,filename+'_w2.csv');
CSVToArray(bias1,filename+'_bias1.csv');
CSVToArray(bias2,filename+'_bias2.csv');
end;
end;
procedure TFSpeech.btn6Click(Sender: TObject);
begin
dlgOpen1.InitialDir:=ExtractFilePath(Application.ExeName)+'Data\NN\';
if dlgOpen1.Execute then
begin
CSVToArray(input_NN,dlgOpen1.FileName);
end;
end;
procedure TFSpeech.btn7Click(Sender: TObject);
var
galat_MSE,err: double;
i,j:Integer;
begin
Label34.Caption:='HASIL = ';
n_input:=Length(input_NN);
n_output:=2;
n_hidden:=50;
SetLength(node1,n_hidden);
SetLength(out1,n_hidden);
SetLength(turunan1,n_hidden);
SetLength(error1,n_hidden);
SetLength(node2,n_output);
SetLength(out2,n_output);
SetLength(turunan2,n_output);
SetLength(error2,n_output);
galat_total:= 1;
iterasi:= 0;
miu:= strtofloat(Edit23.Text);
alfa:= strtofloat(Edit22.Text);
maju;
Edit18.Text:=FloatToStr(out2[0]);
Edit19.Text:=FloatToStr(out2[1]);
galat_MSE:= 1;
pos_MSE:=0;
for i:= 0 to 1 do
begin
err:=0;
for j:= 0 to 1 do
begin
```

```
err:= err+(sqr(dbase_pengenalan[i,j]-out2[j]));
end;
err:= err*0.25;
if err< galat_MSE then
begin
galat_MSE:= err;
pos_MSE:= i;
end;
end;
edit25.Text:= floattostr(galat_MSE);
Label34.Caption:=label34.Caption+dbase_string[pos_MSE];
end;
procedure TFSpeech.timer3Timer(Sender: TObject);
begin
if galat_total> galatmin then
begin
galat_total:= 0;
z:= 0;
while z< Length(erkum) do
begin
maju();
mundur();
update_weight();
galat_total:= galat_total+erkum[z];
series30.AddXY(iterasi,erkum[z]);
z:= z+1;
end;
iterasi:= iterasi+1;
galat_total:= galat_total/length(erkum);
edit12.Text:= floattostr(galat_total);
edit17.Text:= inttostr(iterasi);
end
else
begin
btn3.Caption:= 'start';
timer3.Enabled:= false;
end;
end;
procedure TFSpeech.Button1Click(Sender: TObject);
begin
if button1.Caption='run' then
begin
button1.Caption:='stop';
timer1.Enabled:=true;
timer1.Interval:=1;
alaudioin1.Start;
series1.Clear;
series2.Clear;
nnok:=False;
if cbIden.Checked then
loadWeight;
end
else
begin
button1.Caption:='run';
timer1.Enabled:=false;
alwavelogger1.enabled:=false
end;
end;
procedure TFSpeech.Button2Click(Sender: TObject);
begin
get_trshld:=1;
timer1.Enabled:=true;
timer1.Interval:=1;
```

```
series1.Clear;
series2.Clear;
end;
procedure TFSpeech.TabSheet6ContextPopup(Sender: TObject; MousePos:
TPoint;
var Handled: Boolean);
begin
get_trshld:=0;
alaudioin1.Start;
end;
procedure TFSpeech.Timer1Timer(Sender: TObject);
var
i,j:integer;
max_val,max_valb: double;
tmp,tmp_frek:double;
begin
tmp_frek:=0;
if get_trshld=1 then // ambil threshold
begin
j:=1;
max_val:=0;
max_valb:=0;
while j<100 do
begin
series2.Clear;
Series3.clear;
dft(xt); // dft sinyal vad
i:=0;
while i< 1024 do
begin
series2.AddXY(i,xt[i]);
i:=i+1;
end;
i:=0;
while i<512 do
begin
series3.AddXY(fourier_frek[i],fourier[i]);
i:=i+1;
end;
vad_power[j]:= vad_pow(j);
vad_frekuensi[j]:= vad_frek(j);
if vad_power[j]>max_val then
begin
max_val:=vad_power[j];
vad_power_th:=max_val;
end;
if vad_frekuensi[j]>max_valb then
begin
max_valb:=vad_frekuensi[j];
vad_frekuensi_th:=max_valb;
end;
j:=j+1;
end;
edit2.Text:=floattostr(vad_frekuensi_th);
edit3.Text:=floattostr(vad_power_th);
get_trshld:=0;
timer1.Enabled:= false;
end
else // run
begin
series2.Clear;
Series3.clear;
dft(xt);
i:=0;
```

```
tmp:=0;
while i< 1024 do
begin
series2.AddXY(i,xt[i]); // plot sinyal VAD
tmp:=tmp+sqr(xt[i]); // hit vad power
i:=i+1;
end;
i:=0;
while i<512 do
begin
series3.AddXY(fourier_frek[i],fourier[i]); // plot dft vad
if fourier[i]=1 then
tmp_frek:=i*(sampling/N);
i:=i+1;
end;
if (tmp>vad_power_th)and(tmp_frek>vad_frekuensi_th)and(lala<10240) then
begin
i:=0;
alwavelogger1.FileName:='rekaman.wav';
alwavelogger1.Enabled:=true;
series4.Clear;
series6.Clear;
while i<1024 do
begin
sig[i+lala]:=xt[i];
series4.AddXY(i+lala,sig[i+lala]);
i:=i+1;
end;
fhasil:=FilterButterworth(xt,2000,44100,1);
for I := 0 to high(fhasil) do
begin
sigf[i+lala]:= fhasil[i];
series6.AddXY(i+lala,fhasil[i]);
end;
lala:=lala+1024;
if lala>= 10240 then
begin
alwavelogger1.Enabled:=false;
lala:=0;
series4.Clear;
series6.Clear;
end;
end
else if
(tmp>(vad_power_th*0.65))and(tmp_frek>(vad_frekuensi_th*0.7))and(lala<102
40)
then
begin
i:=0;
while i<1024 do
begin
sig[i+lala]:=xt[i];
series4.AddXY(i+lala,sig[i+lala]);
i:=i+1;
end;
fhasil:=FilterButterworth(xt,2000,44100,1);
for I := 0 to high(fhasil) do
begin
sigf[i+lala]:= fhasil[i];
series6.AddXY(i+lala,fhasil[i]);
end;
lala:=lala+1024;
if lala>=10240 then
begin
```

```
alwavelogger1.Enabled:=false;
lala:=0;
series4.Clear;
series6.Clear;
end;
end
else
begin
alwavelogger1.Enabled:=false;
lala:=0;
end;
dft(sig);
series5.Clear;
i:=0;
while i<512 do
begin
series5.AddXY(fourier_frek[i],fourier[i]);
i:=i+1;
end;
dft(sigf);
series7.Clear;
i:=0;
SetLength(input_NN,512,1);
while i<512 do
begin
series7.AddXY(fourier_frek[i],fourier[i]);
input_NN[i,0]:=fourier[i];
if input_NN[i,0] = 1 then
nnok:=True;
i:=i+1;
end;
lResultIden.Caption:='Hasil Tes = ';
if (cbIden.Checked) and (nnok = True) then
begin
btn7.Click;
lResultIden.Caption :=lResultIden.Caption + dbase_string[pos_MSE];
nnok:=false;
end;
end;
end;
procedure TFSpeech.ALGenericFilter1ProcessData(Sender: TObject;
InBuffer: IALAudioBuffer; var OutBuffer: IALAudioBuffer;
var SendOutputData: Boolean);
var
i: cardinal;
begin
i:=0;
while i<inbuffer.Size do
begin
xt[i]:= inbuffer[i,1];
i:=i+1;
end;
end;
procedure TFSpeech.dft(a: array of double); //a(signal yang mau diolah,
xn hamming atau bukan) b(pola frekuensi) c(amplitude)
var
max_a,min_a: double;
i,j: integer;
begin
i:= 0;
while i< 512 do //pola frekensi ke k
begin
for j:= 1 to 2 do
begin
```

```
ftr[i,j]:= 0;
end;
j:=0;
while j< (N/4) do //x[n]
begin
WijN[i,j,1]:= a[j]*wijn_re(i,j,N);
WijN[i,j,2]:= a[j]*wijn_im(i,j,N);
ftr[i,1]:= ftr[i,1]+ (WijN[i,j,1]); //real xn
ftr[i,2]:= ftr[i,2]+ (WijN[i,j,2]); //imajiner xn
j:= j+1;
end;
fourier[i]:= sqrt((sqr(ftr[i,1]))+(sqr(ftr[i,2])));
fourier_frek[i]:= (i)*(sampling/N);
i:= i+1;
end;
max_a:= 0;
min_a:= 999999999999999;
i:= 0; //untuk normalisasi
while i< (N div 2) do
begin
if fourier[i]> max_a then
begin
max_a:= fourier[i];
end;
if fourier[i]< min_a then
begin
min_a:= fourier[i];
end;
i:= i+1;
end;
i:= 0;
while i< (N div 2) do
begin
if fourier[i]> 0 then
begin
fourier[i]:= sqr((fourier[i]-min_a)/(max_a-min_a));
end
else
begin
fourier[i]:= 0;
end;
i:= i+1;
end;
end;
procedure TFSpeech.FormCreate(Sender: TObject);
begin
if cbIden.Checked then
loadWeight;
end;
end .
```

Halaman ini sengaja dikosongkan

RIWAYAT HIDUP PENULIS

Nama	: Heri Ardiansyah
Alamat	: Jl. Dr. Wahidin SH. No.93, Lamongan
Tempat, Tanggal Lahir	: Bangil, 15 Desember 1980
Agama	: Islam
Status	: Menikah
No. Telp.	: 08563555140
Email	: hery_ardiansyah@yahoo.com
Riwayat Pendidikan	: 1. SDN Kersikan III Bangil
	2. SMPN 1 Bangil
	3. SMAN Bangil
	4. S1 Teknik Elektro Universitas Brawijaya Malang